TECHNISCHE
UNIVERSITÄT
DARMSTADT

Eine vereinheitlichte Modellierungsmethodik für den Entwurf und den virtuellen Test gemischt analog/digitaler Komponenten

Vom Fachbereich Informatik
der Technischen Universität Darmstadt
genehmigte

Dissertation

zur Erlangung des akademischen Grades eines
Doktor-Ingenieurs (Dr.-Ing.)

von

Jürgen Weber, MSc

geboren in Eberbach

Referenten der Arbeit:	Prof. Dr.-Ing. S. A. Huss
	Prof. Dr.-Ing. L. Hedrich
Tag der Einreichung:	14.07.2008
Tag der mündlichen Prüfung:	08.10.2008

Darmstadt 2008, D17

Vorwort

Diese Arbeit ist während meiner Tätigkeit als externer wissenschaftlicher Mitarbeiter am Institut für integrierte Schaltungen und Systeme der Technischen Universität Darmstadt entstanden. Prof. Dr. Sorin A. Huss danke ich hiermit für die ausgezeichnete Betreuung, die Unterstützung bei wissenschaftlichen Publikationen sowie für die vielen Anregungen, Ideen und Diskussionen. Mein herzlicher Dank gilt auch Prof. Dr. Lars Hedrich für die Übernahme des Koreferats sowie sein Interesse an meiner Arbeit.

Besonders bedanke ich mich bei meinen Kollegen Maxim Anikeev, Tom Assmuth, Prih Hastono, Stephan Hermanns, Dan Honciuc, Elisabeth Hudson, Stephan Klaus, Andreas Kühn, Ralf Laue, Felix Madlener, Gregor H. Molter, Tim Sander, Abdul Shoufan, Maria Tiedemann, Song Yuan, Kaiping Zeng und Michael Jung, für die gute Zusammenarbeit, die fachlichen Diskussionen und die Unterstützung vor Ort.

Ich danke der Firma ATMEL GmbH Germany für die Unterstützung meiner Arbeit, sowie für das Bereitstellen wichtiger Tools, die für das Gelingen dieser Arbeit notwendig waren. Weiter danke ich allen Kollegen der CAD-Abteilung und Andreas Lemke, die mich während meiner Arbeit in verschiedenen Bereichen, wie beispielsweise wissenschaftlichen Publikationen, unterstützt haben. Ein besonderer Dank gilt Dr. Mario Anton, der mich seitens der Firma unterstützend durch viele Diskussionen und mit Ideen während meiner Arbeit begleitet hat.

Ebenso haben die durchgeführten Studien- und Diplomarbeiten von den Herren Sven Landa, Peter Stitzelberger und Andrew Burton, wichtige Beiträge für das Entstehen dieser Arbeit geliefert.

Weiter danke ich meinen besten Freunden Jürgen Edelmann und Nicolas König für das Korrekturlesen, sowie für die schöne Zeit am Rande meiner Arbeit.

Mein ganz besonderer Dank gilt meinen Eltern, die diese Ausbildung erst ermöglicht haben. Von ihnen erhielt ich die notwendige Unterstützung im Alltag und jederzeit guten Zuspruch.

Inhaltsverzeichnis

Verzeichnis der Abkürzungen

AC	Alternating Current
AI	Analog Insydes
ASIC	Application Specific Integrated Circuit
AWE	asymptotic waveform evaluation
ATE	Automatic Test Equipment
A/D	Analog/Digital Wandler
BSIM	Spice MOS Device Modell
CAD	Computer Aided Design
CMI	Compiled Model Interface
CAN	Controller Area Network
CMOS	Complementary Metal Oxid Semiconductor
Continuum	Mentor Mixed-Signal Simulator
D/A	Digital/Analog Wandler
DAC	Digital to Analog Converter
DC	Direct Current
DIB	Device Interface Board
DUT	Device Under Test
DSP	Digital Signal Processor
EDA	Electronic Design Automation
EKV	Spice MOS Device Modell (Enz, Krummenacher, Vittoz)
EMV	Elektromagnetische Verträglichkeit
GB	Giga Byte
HDL	Hardware Description Language
IPC	Inter process communication
KOSIM	Simulationswerkzeug der FhG EAS Dresden
MAST	Verhaltensbeschreibungssprache des Simulators Saber
ME	Modelleigenschaft
MOS	Metal Oxid Semiconductor
MOSFET	Metal Oxide Semiconductor Field-Effect Transistor
PLL	Phase-Locked Loop
PV	Prüfvorschrift

RAM	Read Access Memory
Saber	Verhaltenssimulator der Firma Synopsys
SEAP	Simulations-Ergebnis-Analyse-Programm
Spectre	Analogsimulator von Cadence
SpectreVerilog	Mixed-Signal Simulator von Cadence
Spice	Simulation Programm with Emphasis on Integrated Circuits
SPI	Seriell-Parallel Interface
SOI	Silicon on Isolator
TTL	Transistor Transistor Logic
Verilog-A	Analoge Hardwarebeschreibungssprache
Verilog-AMS	Mixed-Signal Hardwarebeschreibungssprache
VeronA	Verifikation analoger Schaltungen (Föderprojekt vom BMBF)
VHDL	Very High Speed Hardware Description Language
VHDL-AMS	VHDL-Analog Mixed-Signal Extension
Verilog-XL	Digitalsimulator von Cadence
VIRTUS	Förderprojekt vom BMBF
VITAL	VHDL Initiative Toward ASIC Libraries
VT	Virtueller Test

Kapitel 1

Einleitung

Moderne Entwurfsmethoden gewinnen heutzutage bei immer komplexer werdenden Mixed-Signal Schaltkreisen mehr und mehr an Bedeutung. Hierzu zählen die Top-Down-Designmethodik, Bottom-Up-Designmethodik oder seit neuesten auch die Meet-In-The-Middel Entwurfsstrategie.

Nicht nur bei der Schaltungsentwicklung werden neue Verfahren eingesetzt, sondern auch in Bereichen wie Testentwicklung, Softwareentwicklung oder übergeordnete Systementwicklung beim Kunden. Alle diese verschiedenen Bereiche versuchen mit neuen Methoden der zunehmenden Komplexität der Systeme entgegen zu wirken und ein verbessertes „time-to-market“ zu erreichen. Hierzu gibt es schon zahlreiche Veröffentlichungen. Einzelne Strategien werden auch in der Industrie erfolgreich angewandt.

Ein wichtiger Bestandteil aller Methoden ist der Einsatz von Verhaltensmodellen, dies einen zusätzlichen Aufwand bedeutet. Bei der Entwicklung dieser Verhaltensmodelle stellt sich meist die Frage der „Wirtschaftlichkeit“, das mehr oder weniger den Erfolg der neuen Strategien in der Industrie bedeutet.

Die Erstellung solcher Verhaltensmodelle spielt somit eine bedeutende Rolle. Lösungen zur optimalen Entwicklung der Modelle gibt es einige. Nachteilig ist, dass diese Modelle in den meisten Fällen speziell nur für einen Bereich (z.B. Systementwicklung) verwendbar sind und keine bereichsübergreifende (z.B. für virtuelle Testsimulationen oder für die Verifikation der Schaltung) Anwendung finden. Dies hängt meist von den unterschiedlichen Modellanforderungen, als auch von der Umgebung, in der das Modell eingesetzt werden soll, (z.B. vom Simulator) ab.

Hier ist beispielsweise im Bereich der Testentwicklung der „Virtuelle Test“ zu nennen, der seit einigen Jahren bekannt ist, aber keinen großflächigen Einzug in der Industrie gefunden hat. Grund hierfür war immer wieder das zusätzliche Erstellen eines DUT-Verhaltensmodells, welches die Forderungen der „virtuellen Testsimulation“ erfüllt, obwohl Verhaltensmodelle, die im Zuge der Schaltungsentwicklung erstellt wurden, vorhanden waren, aber durch die unterschied-

lichen Anforderungen nicht genutzt werden konnten.

Daraus resultiert die Forderung nach der Entwicklung von Modellierungsverfahren, die unter Berücksichtigung des Potentials der verfügbaren Simulationswerkzeuge und HDL-Sprachen eine „entwurfsablaufsübergreifende Modellierung“ mit variablen Modellanforderungen ermöglichen. Speziell für den Bereich Test- und Schaltungsentwicklung werden solche Forderungen immer deutlicher.

1.1 Motivation und Ziel der Arbeit

Ziel dieser Arbeit ist die Schaffung von Methoden und Verfahren zur Erstellung von bereichsübergreifenden Verhaltensmodellen, deren Anforderungen auf den jeweiligen Anwendungsbereich variabel angepasst werden können. Im Fokus dieser Arbeit liegt hierbei das gemeinsame Nutzen von Modellen für die Testentwicklung und für den Designprozess.

Auch innerhalb eines Bereiches z.B. bei der Schaltungsentwicklung müssen je nach Entwurfsstrategie Verhaltensmodelle mit unterschiedlichen Modellanforderungen für jede Designphase erstellt werden. Exemplarisch werden für die Top-Down-Designmethodik Modelle auf unterschiedlichen Abstraktionsebenen benötigt, die durch eine Bottom-Up-Verifikation immer wieder verbessert und angepasst werden müssen. Dieses führt zu einem erhöhten Arbeitsaufwand, den es zu minimieren gilt.

Die Verifikation der Gesamtschaltung auf Transistorebene ist heute bei den meisten Mixed-Signal Schaltkreisen nicht mehr möglich. Es müssen somit immer mehr Verhaltensmodelle eingesetzt werden. Die Anforderungen an ein solches Modell werden durch die jeweiligen Simulationsaufgaben, die für die Verifikation der Schaltung durchgeführt werden müssen, festgelegt. Jede Simulationsaufgabe benötigt, wie in Bild 1.1 ersichtlich, eine oder mehrere bestimmte Modelleigenschaften in dem Verhaltensmodell. Das Problem ist hierbei, dass durch das Implementieren aller Eigenschaften das Modell sehr detailliert und komplex wird, was eine Erhöhung der Simulationszeit zur Folge hat. Deswegen gilt es eine Methode zu entwickeln, die es ermöglicht Modelleigenschaften gezielt für jede Simulationsaufgabe zu aktivieren oder zu deaktivieren.

Ebenso müssen im Bereich der Testentwicklung Modelle mit unterschiedlichen Anforderungen zur Verfügung stehen. Für den virtuellen Test ist es notwendig beispielsweise einige Tests nacheinander simulieren zu können. Hier wird ein DUT-Verhaltensmodell benötigt, dass einen sehr hohen Performancegewinn gegenüber der originalen Simulation auf Transistorebene liefert. Die Modellanforderung liegt nicht nur bei der Geschwindigkeit, sondern auch bei der Genauigkeit. Die Verhaltensmodelle müssen an den zu untersuchenden Stellen im IC die in der Prüfvorschrift definierten Grenzen erfüllen. Von Test zu Test können sich diese Anforderungen immer

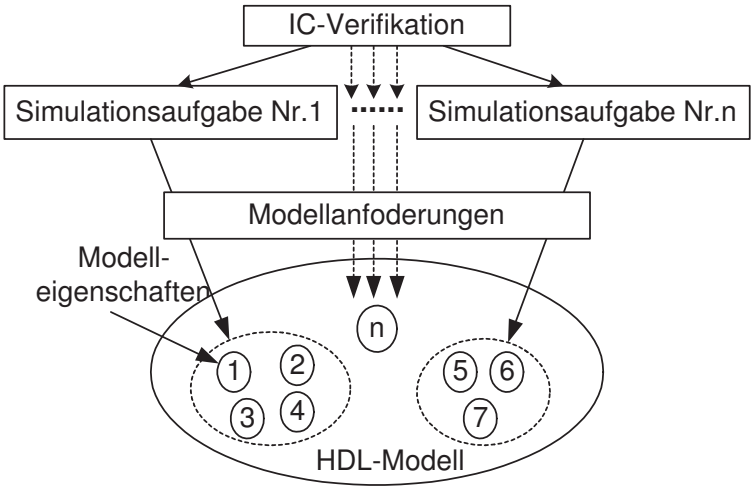

Abbildung 1.1: Simulationsaufgaben bei der Verifikation

wieder verschieben.

1.2 Struktur der Arbeit

Kapitel 2 liefert eine Einführung in die Grundlagen der Verhaltensmodellierung. Hierbei werden verschiedene Abstraktionsebenen, die es bei der Modellierung zu betrachten gilt, vorgestellt. Ebenso werden gängige Entwurfsstrategien, wie die Top-Down-Designmethodik oder die Meet-In-The-Middle Strategie beschrieben. Weiter werden Modellierungsverfahren, die sich in der Praxis schon etabliert haben, erläutert. Fokussiert wird hierbei die „komponentenbasierte Mixed-Level Verhaltensmodellierung“, die ein wichtiges Modellierungskonzept für diese Arbeit darstellt. In einem weiteren Unterkapitel findet sich die DUT-Modellierung ebenso wie die Erläuterungen, was unter „virtuellen Test“ verstanden wird. Hierbei handelt es sich um neue Methoden im Bereich der Testentwicklung, die durch Simulation eine frühzeitige Verifikation des Testprogramms bzw. der Testmethode erhalten. Abgeschlossen wird dieses Kapitel mit einer kurzen Einführung in die Hardwarebeschreibungssprache Verilog-A/AMS, die in dieser Arbeit verwendet wurde. Für die in diesem Kapitel vorgestellten Grundlagen wird der Stand der Technik erläutert und durch Quellen belegt.

Die in dieser Arbeit entwickelte Methodik für die Verhaltensmodellierung wird in Kapitel 3 beschrieben. Beginnend mit einer kurzen Einleitung sowie dem Überblick zum Stand der Technik werden dann Modelleigenschaften und deren Klassifizierung in verschiedene Über- und Untermengen vorgestellt. Diese Eigenschaftsmengen werden folgend als Grundlage für die Erstellung von HDL-Modellen auf verschiedenen Abstraktionsebenen, die als Level1, Level2 und

Level3 bezeichnet werden, verwendet. Neben den verschiedenen Level werden noch Modellvarianten, die unterschiedliche Modelleigenschaften berücksichtigen, eingeführt. Die Mixed-Level Modellierung stellt hierbei die zentrale Modellierungsmethode dar. Weiter werden konfigurierbare HDL-Modelle ergänzend zu den Modell-Level eingeführt, mit dem Ziel Modelleigenschaften variabel einsetzen zu können. Abgeschlossen wird das Kapitel mit einer Erläuterung, wie beide Methoden (Modell-Level und konfigurierbare Modelle) kombiniert in einem Entwurfsablauf eingesetzt werden können.

In Kapitel 4 wird die Realisierung der Modellierungsmethodik erläutert. Hier wird am Beispiel einer Ausgangstreiberstufe die Erstellung der verschiedenen Level-Modelle demonstriert. Weiter werden die Realisierungsmöglichkeiten für konfigurierbaren HDL-Modelle und deren Vor- und Nachteile in der HDL Verilog-A/AMS vorgestellt. Die Entwicklung von konfigurierbaren MOSFET HDL-Modellen in Verilog-A wird ebenfalls in diesem Kapitel beschrieben. Diese ermöglichen es, verschiedene Funktionen, die ein MOSFET-Modell beinhaltet, gezielt zu aktivieren oder zu deaktivieren mit dem Ziel ein reduziertes Gleichungssystem zu erhalten. Angewandt wurde das MOSFET HDL-Modell an der Ausgangstreiberstufe, die durch eine Mixed-Level Modellierung entstanden ist. Eine kurze Beschreibung mit einem Beispiel zur Bestimmung der Modellgenauigkeit schließt dieses Kapitel ab. Hier werden kurz die Ermittlungen der Modellabweichungen, die in der gesamten Arbeit verwendet werden, demonstriert.

Im vorletzten Kapitel werden die in dieser Arbeit vorgestellten Methoden anhand eines realen Beispiels für die Schaltungsentwicklung und für Simulationen aus der Testentwicklung gezeigt. Bei dem Demonstrator handelt es sich um einen Antennentreiber für niedrige Frequenzen. Er treibt Antennen, die in automatischen Systemen für die Fahrzeugschlossentriegelung (Passive Entry/Go Systeme) verwendet werden. Stellvertretend für alle Komponenten des Schaltkreises wird der Aufwärtswandler, der als Funktionseinheit im IC vorhanden ist, näher vorgestellt. Die HDL-Modelle des Wandlers werden einmal für die Anwendungen des virtuellen Tests und zum andern für die Anforderungen bei der Schaltungsentwicklung aufbereitet. Danach werden Ergebnisse des Gesamtsystems sowohl für die Schaltungsverifikation in der Schaltungsentwicklung als auch für Testsimulationen für die Testentwicklung präsentiert. Am Ende des Kapitels werden die Ergebnisse kurz zusammengefasst.

Abgeschlossen wird diese Arbeit in Kapitel 6 mit einer Zusammenfassung und einem Ausblick.

Im Anhang wird detaillierter auf den virtuellen Test und dessen Stand der Technik eingegangen. Weiter befinden sich dort Grundlagen zum MOSFET Level1-Modell und eine ausführliche Beschreibung zu den hier in der Arbeit verwendeten Bewertungsmethoden.

Kapitel 2

Grundlagen der Verhaltensmodellierung

Bei dem heutigen Entwurf von immer komplexer werdenden Mixed-Signal Schaltkreisen, gewinnt die Verhaltensmodellierung mehr und mehr an Bedeutung. Die Verifikation eines Gesamtsystems ist auf Transistorebene fast nicht mehr möglich. Ebenso bei neuen Entwurfsmethoden wie z.B. der Top-Down-Methode ist die Modellierung ein fester Bestandteil der Entwurfsstrategie.

In modernen Designflows für Mixed-Signal Systeme wird in der Regel eine Top-Down-Designmethodik angestrebt. Ausgehend von einer abstrakten Beschreibung des Gesamtsystems werden bis herunter zur finalen Realisierung auf Transistorebene kontinuierlich Verfeinerungen vorgenommen. Eine Systembeschreibung auf einer hohen Abstraktionsebene kann z.B. mit Hilfe von Sprachen wie VHDL-AMS oder Verilog-AMS vorgenommen werden und dient in der Regel dazu, die in der Spezifikation festgehaltenen Systemeigenschaften mit Hilfe von Simulationen zu überprüfen. Ein mit diesem Ziel erstelltes Verhaltensmodell kann im erweiterten Sinne auch als ausführbare Spezifikation gesehen und mit dem Kunden ausgetauscht werden.

Die im Zuge einer Top-Down-Entwurfsmethodik [Kun04] vorgenommenen Verfeinerungen der Systembeschreibung berücksichtigen auf unteren Entwurfsebenen zunehmend Implementierungsaspekte. Wird zunächst nur die reine Funktion betrachtet, rückt die Fragestellung mit welchen schaltungstechnischen Ansätzen diese Funktion realisiert werden kann immer mehr in den Vordergrund. Auf möglicherweise unterschiedlichen Hierarchieebenen [Hus01A] wird mit der endgültigen Transistornetzliste Funktions- und Pinkompatibilität hergestellt. Eine sinnvolle Erweiterung der Top-Down-Entwurfsmethodik besteht in der Bottom-Up-Verifikation [Eck00]. Auf der Basis von Charakterisierungsergebnissen, die mit Hilfe von Simulationen oder Messungen gewonnen werden können, werden die Modellbeschreibungen kalibriert.

Trotz naheliegender Vorteile hat sich die Top-Down-Designmethodik in der Praxis noch nicht vollständig durchgesetzt. Dies liegt zum einen an der noch fehlenden Akzeptanz von Mixed-

Signal Hardwarebeschreibungssprachen, verglichen mit rein digitalen Sprachen wie VHDL und Verilog, deren Nutzung in der Digitaltechnik inzwischen zum Standard geworden ist. Zum anderen existieren noch keine vereinheitlichten Standards für den Übergang der Modelle zwischen den einzelnen Hierarchieebenen. Hier ist insbesondere die Fragestellung hervorzuheben, welche Funktionen auf welchen Hierarchieebenen sinnvollerweise zu beschreiben sind und wie die Modellschnittstellen in diesem Kontext zu behandeln sind.

Demnach wird ein Großteil der Schaltungen noch immer mit Hilfe der Bottom-Up-Entwurfsmethodik realisiert. Hier werden zunächst die Einzelkomponenten des Gesamtsystems auf Transistorebene realisiert, anschließend zu größeren Einheiten verschaltet und mit Hilfe von Simulationen verifiziert. Ein Risiko der Bottom-Up-Methodik ist, dass mögliche Fehler in der Systemkonzeption erst spät im Designablauf aufgedeckt werden können und u.U. aufwendige Korrekturmaßnahmen bedürfen [Ant03].

Werden Verhaltensmodelle und Transistornetzlisten bei der Simulation gemischt, so wird dies als Mixed-Level Simulation bezeichnet. Derartige Ansätze werden i.d.R. bei Gesamtsystemsimulationen herangezogen, um durch den Austausch rechenzeitintensiver Komponenten Simulationszeit einzusparen [Web05].

Mixed-Level Simulationen können grundsätzlich unabhängig von der verwendeten Entwurfsmethodik verwendet werden. Insbesondere im Umfeld von Automotive Schaltkreisen werden die Hersteller immer häufiger mit der Forderung des Kunden nach der Bereitstellung von Verhaltensmodellen konfrontiert. Wurden diese Modelle bereits im Rahmen des Entwurfsprozesses realisiert und müssen nicht nur aufgrund des Kundenwunsches entwickelt werden, stellt dies eine erhebliche Arbeitseinsparung dar.

Eine ähnliche Situation herrscht in Bezug auf den virtuellen Test, mit dem eine weitere Verkürzung der Entwurfszeiten angestrebt wird. Durch die Bereitstellung von Verhaltensbeschreibungen des Schaltkreises und des Loadboards kann mit der Erstellung des Testprogramms begonnen werden, bevor Silizium vorliegt, d.h. es kann eine Parallelisierung von Schaltungs- und Messtechnikentwicklung vorgenommen werden. Idealerweise sollten Verhaltensmodelle für den virtuellen Test aus bereits im Entwurfsprozess realisierten Modellbeschreibungen abgeleitet werden.

Die Anforderungen an ein Modell können sehr unterschiedlich von dem Abstraktionsgrad bis hin zur Genauigkeit und der Performance sein. Die Grundlagen der Verhaltensmodellierung sowohl im Entwurfsprozess als auch für den virtuellen Test sollen in diesem Kapitel erläutert werden. Weiter wird der „Stand der Technik“ in den einzelnen Abschnitten vorgestellt.

2.1 Abstraktionsebenen

Eine Schaltung kann unterschiedlich detailliert in einer HDL beschrieben werden. Der Grad der Detaillierung wird mit Abstraktionsgrad bzw. Abstraktionsebene bezeichnet. Hierbei stellt eine funktionale Beschreibung ein sehr hohen, und die Beschreibung mittels Basiselemente z.B. Transistormodellen einen sehr niedrigen, Abstraktionsgrad dar. Bei einem Top-Down-Design werden in der Entwurfsphase alle Abstraktionsebenen von der funktionalen Beschreibung bis hin zu Layout durchlaufen. Hierbei wird der Übergang von einer höheren auf eine niedrigere Ebene als Implementierung beschrieben. Hingegen wird bei einer Bottom-Up-Verifikation das Wechseln von einer niedrigeren auf eine höhere Ebene als Abstraktion bezeichnet.

Im Designprozess gibt es neben der strukturellen Sicht auch noch die Sicht auf die Geometrie und das Verhalten. Dies wird in der Literatur am besten durch das Y-Diagramm nach Gajski-Walker [Wal85] beschrieben.

Die Definition einheitlicher Abstraktionsebenen für Analog- und Digitalschaltungen ist in der Literatur selten zu finden. Aus diesem Grund wurden für den Digital- und Analogentwurf unterschiedliche Ordnungen der Abstraktionsebenen eingeführt.

2.1.1 Abstraktionsebenen für den Digitalentwurf

Die Abstraktionsebenen für den Digitalentwurf sind in einem lang andauernden Entwicklungsprozess entstanden. Die Definition der verschiedenen Abstraktionsebenen ist anerkannt und weit verbreitet. Im nachfolgenden Diagramm sind die verschiedenen Ebenen dargestellt [Hus01A].

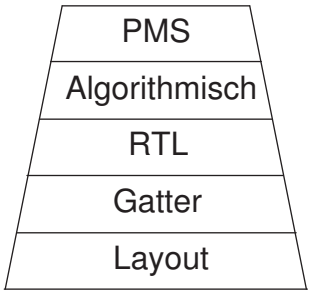

Abbildung 2.1: Abstraktionsebenen für den Digitalentwurf

- PMS
- Schaltkreis (Algorithmische Ebene)
- Registertransfer

- Gatter (Logik-Ebene)
- Schaltung (Transistorebene)
- Layout

Die **Layoutebene** repräsentiert die physikalische Realisierung der Schaltung als Layout. Auf der **Schaltkreis-** oder auch **Transistorebene** werden die Bauteile mit algebraischen Gleichungen, oft in der Form von Differenzialgleichungen (siehe Anhang B), beschrieben. Der Schaltkreis besteht aus aktiven und passiven elektrischen Bauelementen, die im Zeitbereich beschrieben werden. Auf **Gatter-** oder auch **Logikebene** erfolgt die Beschreibung einer Schaltung durch logische Verknüpfungen. In der Verhaltenssicht erfolgt dies durch boolesche Gleichungen und in der strukturellen Sicht durch eine Zusammenschaltung von logischen Gattern und Speicherelementen z.B. FlipFlops.

Register Transferebene:
Die Beschreibung erfolgt durch Operationen (z.B. Addition) und durch den Transfer der verarbeiteten Daten zwischen Registern. Strukturell erfolgt dies durch Blöcke wie Register, Zähler, Multiplexer und Speichermodule. Die Spezifikation erfolgt durch erweiterte Boolesche Gleichungen, Wahrheits- oder Zustandsübergangstabellen, Mikrooperationen und Signalzuweisungen.

Algorithmischer Ebene:
Die Spezifikation auf algorithmischer Ebene erfolgt durch nebenläufige notierte Algorithmen. Hier wird typischerweise noch keine Rücksicht auf eine eventuelle Struktur genommen. Typische Strukturelemente auf dieser Ebene sind Mikroprozessoren, Kanalwerke, Speicherbänke und Bussysteme.

PMS (Processor, Memory, Switch):
Die strukturelle Beschreibung erfolgt durch Prozessoren, Speicherbänke und Busssysteme, die den Entwurf von Rechnerarchitekturen ermöglichen.

2.1.2 Abstraktionsebenen für den Analogentwurf

Die Definition der verschiedenen Abstraktionsebenen im Digitalen hat sich weitgehend etabliert, während in der analogen Welt keine genaue Festlegung der Abstraktionsebenen und der Transformation zwischen den einzelnen Ebenen besteht. Bild 2.2 stellt eine möglich Abstraktionsebenen

Hierarchie dar (siehe [Hus01A]). Weitere ähnlich Ansätze können in [Mos97] nachgelesen werden.

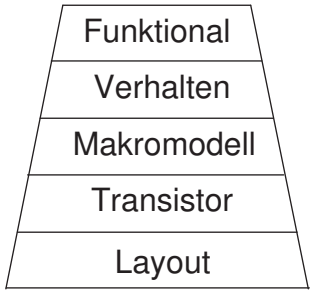

Abbildung 2.2: Abstraktionsebenen für den Analogentwurf

- Funktionale Ebene
- Verhaltensebene
- Makromodellebene
- Transistorebene
- Layoutebene

Layoutebene:
Auf der Layoutebene ist die physikalische Realisierung der Schaltung als Layout festgelegt.

Transistorebene:
Auf Transistorebene werden aktive und passive Bauelemente als Grundelemente verwendet, die mit algebraischen Gleichungen, oft in der Form von Differenzialgleichungen, beschrieben werden.

Makromodellebene:
Makromodelle sind im engeren Sinne Schaltungen, die mit den Standardelementen vom Schaltungssimulatoren wie z.B. Spectre auskommen. Die Modellierungsmethode besteht darin ideale Funktionsblöcke, die mit linearen oder nichtlinearen Gleichungen beschrieben sind, so zu verwenden, dass die gewünschte Modellfunktion erfüllt werden kann. Weiter werden z.B. Spice

Ersatzschaltungen wie ein OP als Makromodell bezeichnet. Im allgemeinem Sinne können Makromodelle aus einer Kombination von Simulatorstandardelementen und eigenen Verhaltensmodellen, die in einer HDL beschreiben sind, bestehen.

Verhaltensebene:
Modelle auf der Verhaltensebene geben die Sicht von außen auf das zu modellierende System wieder. Diese werden durch mathematische Funktionsgleichungen zwischen den Ein- und Ausgangssignalen beschrieben. Es handelt sich hierbei um konservative Klemmen, die dem Gesetz der Energieerhaltung unterliegen. Bei elektrischen Schaltkreisen sind dies Strom und Spannung, die nach den Gesetzen von Kirchhoff beschrieben werden.

Funktionale Ebene:
Im Gegensatz zur Verhaltensebene gelten die Energieerhaltungssätze auf der funktionalen Ebene nicht mehr. Auf dieser Ebene werden komplexe Funktionsblöcke mit mathematischen Funktionen beschrieben. Diese funktionalen Einheiten kommunizieren miteinander und bilden so das beabsichtigte Verhalten des Systems wieder. Innerhalb der Systembeschreibung gibt es keine Signale mehr, die eine direkte physikalische Interpretation aufweisen. Die Beschreibung beschränkt sich auf die Leistungsmerkmale und die grundlegenden Charakteristika.

2.1.3 Abstraktionsebenen für den Mixed-Signal Entwurf

Im Bereich Mixed-Signal gibt es noch weniger einheitliche Definitionen der verschiedenen Abstraktionsebenen. Das Problem ist, dass eigentlich sowohl die digitalen als auch die analogen Abstraktionsebenen ihre Berechtigung in einem Mixed-Signal Design haben, aber eine Vereinigung sehr schwer ist. Ein weiteres Problem stellt die Definition der Verhaltensmodelle dar. Diese können auf mehreren Abstraktionsebenen verwendet werden und nicht nur auf der Verhaltensebene. Trotz dieser Probleme gibt es einige Ansätze in der Literatur, die Abstraktionsebenen für den Mixed-Signal Bereich definieren, wie in [Hus01A]. Hier sind die verschiedenen Ebenen übergeordnet in zwei Kassen eingeteilt: in funktionale Modelle und in Verhaltensmodelle. Die funktionalen Modelle beinhalten den **Konzeptionellen** und den **Analytischen Bereich**. Bei den Verhaltensmodellen ist der **Algorithmische Bereich**, sowie die **Prozedurale** und die **Komponenten Ebene** zu finden (siehe Bild 2.3).

Der **Konzeptionelle Bereich** soll hierbei die abstrakteste Ebene darstellen, in der sowohl digitale abstrakte Grundfunktionen als auch die analogen Rechenmethoden ihre Anwendung finden. Zur Zeit gibt es hier aber noch keine genauen Vorstellungen wie diese abstrakte Beschrei-

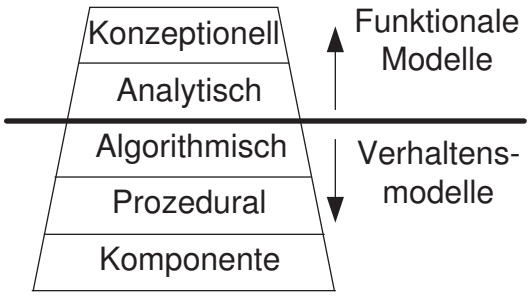

Abbildung 2.3: Abstraktionsebenen für den Mixed-Signal Entwurf

bungen im Mixed-Signal Bereich aussehen können.

In dem **Analytischen Bereich** wird die Funktionalität des Schaltkreises mit idealisierten analytischen Gleichungen für kontinuierliche Signale beschrieben, bei dem die kausale Modellierung zu Grunde liegt.

Für den **Algorithmischen Bereich** werden Verhaltensmodelle verwendet, die mit einem nichtkausalem Modellierungskonzept für kontinuierliche Signale beschrieben werden und somit den Energieerhaltungssätze unterliegen. Weiter können die ereignisgesteuerten Prozessalgorithmen sowohl mit kausalen als auch mit nichtkausalen Modellierungsarten beschrieben werden.

Die **Prozedurale Ebene** adressiert Verfeinerungsoperationen von Modellen, die auf der Algorithmischen Ebene entstanden sind. Dies kann sowohl „building“ Blocks als auch Prozesseinheiten beinhalten. Die Makroebene im Analogen ist eine Untermenge der Prozeduralen Ebene.

Auf der **Komponenten Ebene** werden Grundelemente, die in einer Bibliothek für eine Technologie zur Verfügung stehen, verwendet. Diese Elemente werden in einer strukturellen Netzliste miteinander verbunden.

2.2 Entwurfsstrategien

Heutzutage werden verschiedene Entwurfsstrategien im Entwicklungsprozess angewandt. In diesem Kapitel sollen die Strategien vorgestellt werden, die immer mehr an Bedeutung bei den heutigen Entwicklungen von Mixed-Signal ICs gewinnen. Hierzu zählt, die schon seit einigen Jahren publizierte Top-Down-Designmethodik, welche aber erst in den letzten Jahren mit steigender Tendenz zum Einsatz kommt. Grund hierfür sind die seit kurzem verfügbaren standardisierten Mixed-Signale Sprachen wie VHDL-AMS oder Verilog-AMS, die auch von den Simulatoren im Designflow unterstützt werden. Weiter soll die Meet-In-The-Middle Entwurfsstrategie, die auch bei dem in dieser Arbeit präsentierten Demonstrator verwendet wurde, vorgestellt werden.

2.2.1 Top-Down-Designmethodik

Der Entwurf immer komplexer werdender Systeme kann nicht mehr mit herkömmlichen Designmethoden gelöst werden. Ebenso ist eine Verifikation durch Simulation auf Transistorebene mit vertretbarem Zeitaufwand nicht mehr durchführbar. Ein strukturiertes Top-Down-Design ([Kun00], [Heu98A], [Heu98B]) mit anschließender Buttom-Up-Verifikation [Eck00] ist ein wesentliches Hilfsmittel das anstehende Komplexitätsproblem zu lösen. Ziel des Top-Down-Design ist es, von einer abstrakten Beschreibung des Verhaltens über eine detaillierte Beschreibung der Struktur (z.B. Netzliste) auf eine geometrische Beschreibung (Layout) für die Fertigung zu gelangen.

Der Startpunkt eines Top-Down-Designs ist die Beschreibung des Schaltkreises auf einer hohen Abstraktionsebene auf der Basis der Spezifikation. Sollte es sich um ein sehr komplexes Design handeln, ist es ratsam zuvor noch eine Partitionierung des Schaltkreises vorzunehmen. Je nach Schaltungsart kann die oberste Abstraktionsebene sowohl eine Funktionale Ebene (z.B. Matlab, System-C, Datenfussdiagramme ...), die alle nichtkonservative Systeme beinhaltet, als auch eine Verhaltensebene (Verilog-AMS, VHDL-AMS), welche die Energieerhaltungssätze berücksichtigt, sein [Jor01].

Ausgehend von dieser Abstraktionsebene wird in der Entwurfsphase die nächst niedrigere Ebene entwickelt bis hin zur Komponenten bzw. Physikalischen Ebene. Hierbei ist es nicht zwingend notwendig alle Ebenen zu durchlaufen. Es kann durchaus sein, dass bei manchen Schaltungsklassen z.B. eine Makromodellierung keinen Sinn macht, oder Modellierungsmethoden verwendet werden, die eine Mischung von Verhaltensmodellen und Makromodellen beinhalten (siehe Mixed-Level Modellierung).

Ein weiterer wichtiger Punkt des Top-Down-Design ist die Austauschbarkeit der Schaltungsblöcke der verschiedenen Abstraktionsebenen. Nach jeder Realisierung eines Schaltungsblockes auf einer niedrigeren Abstraktionsebene, sollte dieser zur Überprüfung seines Verhaltens in dem Gesamtsystem auf einer höheren Ebene eingebunden werden. Ein weiterer Grund für die Austauschbarkeit der Blöcke ist die finale Gesamtsimulation des Schaltkreises. Hier ist es wünschenswert Verhaltensmodelle einzusetzen, die im Zuge der Top-Down-Methodik entstanden sind. Die Forderung nach der Austauschbarkeit und somit auch der Pinkompatibilität ist sehr schwer realisierbar. Erste Vorschläge sind in [Jor01] beschrieben.

Ein weiterer Vorteil der Top-Down-Designmethodik ist die frühzeitige Bereitstellung der Verhaltensmodelle für den virtuellen Test.
Die Forderung der Kunden nach einer „simulierbaren Spezifikation“, um deren Applikation bzw. Gesamtsystem zu überprüfen, wird somit auch erfüllt.

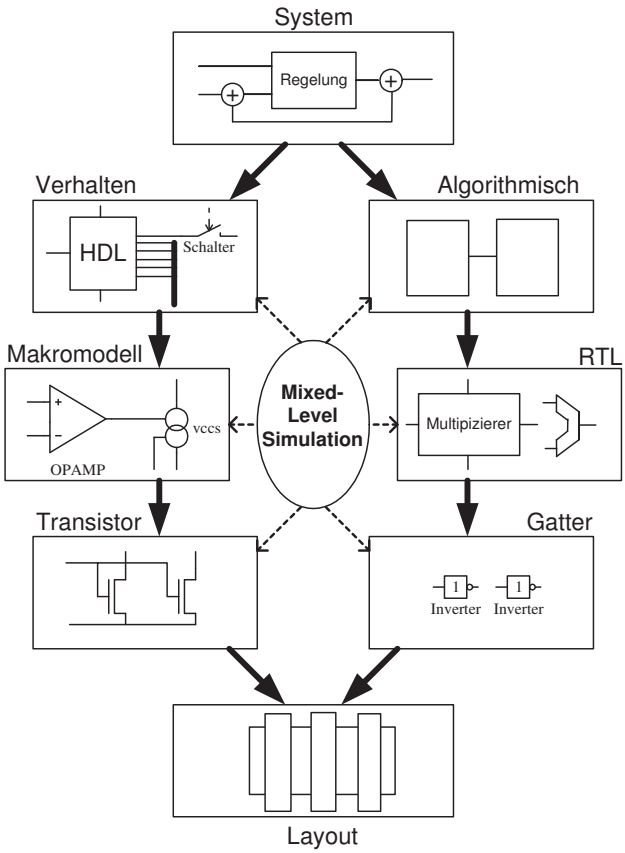

Abbildung 2.4: Top-Down-Designmethodik

2.2.2 Meet-In-The-Middle

Eine wichtige Entwurfsstrategie ist der Ansatz von Meet-In-The-Middle [Eva91], der sowohl die Top-Down als auch die Bottom-Up-Strategie in sich vereint. Bei diesem Ansatz werden zunächst parametrisierbare Grundschaltungen auf der Basis der technologischen Realisierbarkeit mit einer Bottom-Up-Strategie entworfen. Auf Basis dieser ermittelten realisierbaren Basiskomponente, erfolgt dann Top-Down der Entwurf unter Berücksichtigung der Soll-Funktionalität des Designs.

Nicht nur im Mikrosystementwurf [Mue94] wird die Meet-In-The-Middle Strategie eingesetzt, sondern auch beim Entwurf analogen bzw. Mixed-Signal Schaltungen ([Hed03], [Mar08]). Ausgehend von den Eigenschaften der Technologie in der entwickelt werden soll, werden die im Design identifizierten kritischen Komponenten zuerst modelliert und entwickelt. Ist diese Hürde genommen, kann das komplette Design im Zuge einer Top-Down-Methode realisiert werden.

2.3 Modellierungsverfahren

In diesem Abschnitt werden zunächst einige grundlegende Begriffe eingeführt und unterschiedliche Methoden der Modellentwicklung vorgestellt. Ein besonderer Fokus liegt hierbei in der „Komponentenbasierten Mixed-Level Modellierung“, die im Rahmen dieser Arbeit häufig Anwendung findet. Bei der Bottom-Up-Methode gibt es verschiedene Möglichkeiten ein Modell zu generieren. Hierzu zählt beispielsweise die **symbolische Analyse** [Bor98]. Ein Anwendungstool für diese Analyse ist z.B. Analog Insydes [Hal02]. Bei diesen Verfahren werden aus einer Netzliste Gleichungen generiert, die im nächsten Schritt vereinfacht werden. Mit diesen vereinfachten Gleichungen kann mit Hilfe einer Verhaltensmodellierungssprache eine Komponente beschrieben werden. Von Nachteil ist, dass z.Z. diese Methode nur im Kleinsignalbereich (AC) und für Schaltungen geringer Komplexität (max. 20 Elemente) sinnvoll eingesetzt werden kann. Weiter können eingeschränkt nichtlineare Effekte (DC) berücksichtigt werden, was jedoch zur Folge hat, dass das Gleichungssystem sehr groß werden kann. Im Grossignalbereich sind erste Ansätze vorhanden, die aber für die Generierung von „schnellen“ Verhaltensmodellen, nur eingeschränkt geeignet sind.

Eine andere Methode zur Modellerstellung ist die **blockorientierte Modellierung**. Hierzu zählt beispielsweise das Prinzip der Makromodellierung. Einer der Vorteile besteht darin, dass keine Kenntnis der internen Struktur des Bauelementes erforderlich ist, da nur das Verhalten und nicht der reale Aufbau nachgebildet wird. Dadurch sind die Modelle einfacher und benötigen weniger Rechenzeit, so dass auch komplexe Schaltungen simuliert werden können [Mam96].

Im Förderprojekt HDL-VMS [Eck99] wird eine weitere blockorientierte Methode zur Generierung von Verhaltensmodellen mit Hilfe eines Bibliothekskonzeptes vorgestellt. Untersucht man bestehende Mixed-Signal Schaltungen nach den in ihnen vorkommenden analogen Schaltungsblöcken geringer Komplexität, so werden folgende Punkte deutlich:

- Der Großteil der analogen Schaltungen lässt sich wenigen Schaltungsklassen zuordnen.
- Mit wenigen Schaltungsstrukturen können viele Anwendungsfälle einer Schaltungsklasse abgedeckt werden.
- Eine gegebene Schaltungsstruktur wird fast nie mehrmals mit identischer Dimensionierung verwendet.

Eine detailliertere Beschreibung der Anforderungen an eine Bibliothek analoger Schaltungen sind in [Eck99] näher beschrieben.

Eine weitere Modellierungsmethode stellt die Nachbildung von Kennlinien dar. Bauelemente und Komponenten werden häufig durch verschiedene Kennlinien charakterisiert, die unter-

einander in Abhängigkeit stehen können. Messergebnisse oder Ergebnisse aus der Simulation können ebenfalls in Kennlinienform dargestellt werden. Um diese zum Teil mehrdimensionale Kennlinien in einem Verhaltensmodell nutzen zu können, muss aus ihnen eine funktionale Beschreibung erzeugt werden [Kem96]. Es gibt zahlreiche mathematische Verfahren, mit denen Funktionsgraphen, Kennlinien oder Datentabellen durch einfache Funktionen in geschlossener Form angenähert werden können. Zu unterscheiden ist in der Regel zwischen zwei Verfahren, der Interpolation und der Approximation. Bei der Approximation [Schw98] werden Werte aus der Datentabelle durch eine vorgegebene funktionale Abhängigkeit bestmöglich angenähert. Dabei werden folgende Verfahren unterschieden:

- Polynomfit (z.B. Gerade oder Parabel als Ausgangspolynom)
- Linearkombination beliebiger Funktionen
- Nichtlineare Ausgleichsrechnung

Aufgabe der Interpolationsrechnung ist es, zu gegebenen Punktepaaren (x_i, y_i) eine geeignete Interpolationsfunktion Y(x) zu finden, die durch die Punktepaare hindurch verläuft. Hierzu wurden einige Tools entwickelt die es ermöglichen, aus charakterisierten Kennlinien eine Funktionsbeschreibung zu generieren (z.B. GenF [Kem96], [Ros01]).

Eine weitere Möglichkeit besteht darin, eine nichtlineare analoge Schaltung abschnittsweise linear zu beschreiben. Diese Methode wird mit dem Werkzeug AWE (asymptotic waveform evaluation) realisiert [Rag93].

Weitere Ansätze sind:

- Graphbasierte Modellierung [Gri96]
- Hybride Datenflussgraphen [Ley95]
- Hybride Petri-Netze [Ste93]
- Empirische Verfahren [Box87]
- Systemtheoretisch basierte Methoden [Prä91]

2.3.1 Komponentenbasierte Mixed-Level Modellierung

Wie in der Einleitung erläutert, werden im Rahmen des integrierten Schaltungsentwurfs Verhaltensmodelle in unterschiedlichen Anwendungsbereichen benötigt. Da sich noch keine vereinheit-

lichten Modellierungsstrategien etabliert haben, die den sämtlichen Anwendungsbereichen (ausführbare Spezifikation, Top-Down- und Bottom-Up-Methodik, Kundenmodelle und Virtueller Test) gerecht werden, müssen anwendungsspezifische Lösungen mit größtmöglicher Abdeckung der unterschiedlichen Anforderungen angestrebt werden.

In diesem Abschnitt soll ein Modellierungskonzept vorgestellt werden, welches die effiziente Realisierung von Modellen auf Komponentenebene ermöglicht. Die Grundidee dieses Ansatzes liegt darin, das Konzept der Mixed-Level Simulation auch auf Komponentenebene anzuwenden. So besteht die Möglichkeit, an geforderten Stellen das Schaltungsverhalten hochgenau zu beschreiben, zusätzlich jedoch mit gezielten Vereinfachungen die Simulationszeit erheblich zu reduzieren. Hohe Modellgenauigkeit wird in der Regel an den Schnittstellen zur umgebenden Schaltung, des Loadboards oder der Applikation gefordert. Im Falle von Kundenmodellen wird z.B. oftmals die exakte Abbildung der Ausgänge von Treibern und Versorgungsstufen verlangt, um das Zusammenspiel des ICs mit der Applikation untersuchen zu können. Beim virtuellen Test müssen sogar teilweise die Spannungsabfälle der Dioden der ESD-Strukturen in den Modellen nachgebildet werden [Ant02]. Ausgehend von diesen Anforderungen basiert der in Kapitel 3 präsentierte Modellierungsansatz darauf, pin- und funktionskompatible Modellbeschreibungen zu gewinnen, indem Subblöcke der Transistornetzliste durch HDL-Beschreibungen ausgetauscht werden. In diesem Zusammenhang sei darauf hingewiesen, dass im Zuge einer

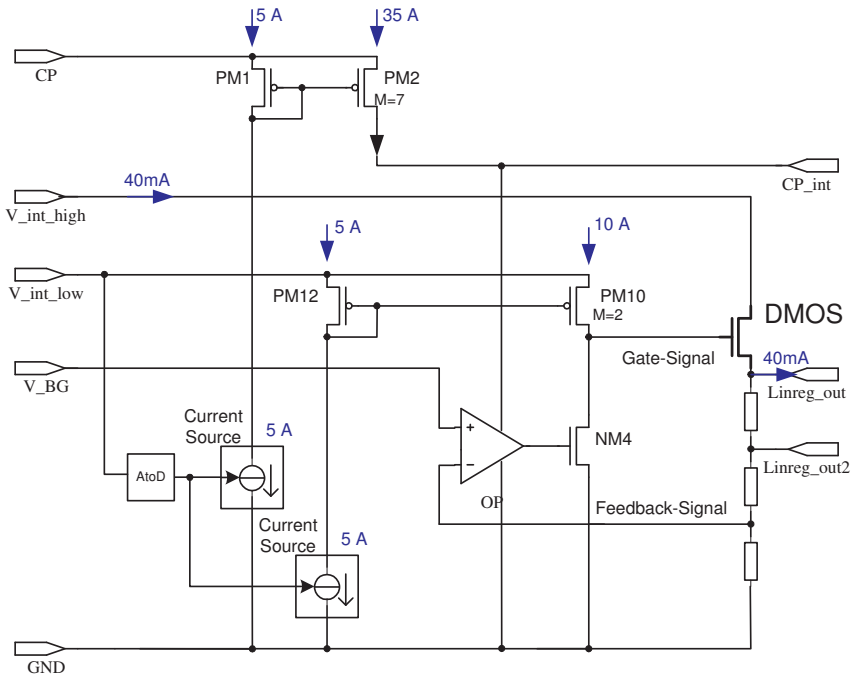

Abbildung 2.5: Beispiel Spannungsregler

Top-Down-Methode, pinkompatible Modelle nur dann erstellt werden können, wenn weitgehende Kenntnisse über die Implementierung der Schaltung vorliegen. Von daher existieren zumeist nur vereinfachte Beschreibungen auf höheren Hierarchieebenen, die dennoch wertvolle Hinweise auf zu vereinfachende Schaltungsteile oder Klemmen liefern, die mit Hilfe digitaler Signale beschrieben werden können. Dieser Modellierungsansatz ist an die Verfügbarkeit geeigneter Simulationswerkzeuge gebunden. Simulationswerkzeuge wie ADVanceMS, AMS-Designer oder Smash unterstützen die Forderung der Instanziierung von HDL-Code aus Netzlisten und umgekehrt jedoch uneingeschränkt. In Bild 2.5 ist ein Spannungsregler mit diesem Modellierungsansatz entstanden. Eine ausführliche Beschreibung dieses Ansatzes kann in [Web05] nachgelesen werden. Dieser Ansatz findet in dieser Arbeit sehr häufig seinen Einsatz.

2.4 DUT-Modellierung

Eine Mixed-Signal Prüfung eines kompletten ICs beinhaltet im Allgemeinen einige hundert Tests und dauert am Tester je nach Schaltungstyp- und Komplexität von einigen Millisekunden bis einigen Sekunden. Dies ist im Vergleich zu den zur Schaltungsverifikation verwendeten Simulationsintervallen eine extrem lange Zeit. Bei den immer umfangreicher werdenden Mixed-Signal Schaltkreisen ist es in den meisten Fällen deshalb gar nicht mehr möglich eine Gesamtsimulation auf Transistorebene durchzuführen.

Es ist daher absolut notwendig schon bei der Verifikation der Gesamtschaltung Verhaltensmodelle zu verwenden. Bei der Gesamtsimulation sind in der Regel die einzelnen Subblöcke schon für sich getestet, damit nur das Zusammenspiel aller Komponenten verifiziert werden muss. Die Anforderungen an die Verhaltensmodelle sind so ausgerichtet, dass eine Gesamtsimulation im Rahmen einiger Stunden ablaufen sollte. Meist wird die Simulation für das Betrachten weniger Funktionalitäten des Schaltkreises aufgesetzt. Dies wäre in etwa mit einem Test am realen Tester vergleichbar, was zur Folge hätte, dass bei mehreren hunderten Tests die Simulationszeit inakzeptabel wäre. Demzufolge müssen die Anforderungen an ein Verhaltensmodell sowohl in Richtung Performance als auch an manchen Stellen in Richtung Genauigkeit steigen.

Im Förderprojekt VIRTUS [Sax00] wird darauf hingewiesen, dass der Aufwand für diese Modellierung erheblich und grundsätzlich eine Automatisierung nicht möglich ist. Weiter wird festgestellt, dass die Modellverifikation in der Regel ein schwieriges Problem darstellt [Ein95]. Die Modellierung kann erst erfolgen, wenn der Schaltkreis weitgehend abgeschlossen ist. Das Ziel, mit der Testprogrammverifikation vor dem ersten Silizium beginnen zu können, ist somit nur schwer zu erreichen. Im genannten Projekt werden für komplexere Schaltkreise zur Spezifikations- und Gesamtverifikation Systemmodelle verwendet. Diese sind abstrakte Model-

le, welche die Funktionalität, aber nicht die spätere Realisierung widerspiegeln. Die Simulationsintervalle können bis zu einigen Sekunden betragen, wobei die Rechenzeiten ebenfalls nur im Stundenbereich liegen sollen. Diese hohe Simulationsgeschwindigkeit wird durch einfachere Simulationsalgorithmen und die abstrakte Beschreibung erreicht. Damit ist aber die erreichbare Genauigkeit begrenzt.

Um aus einem Systemmodell ein für die Testsimulation verwendbares DUT-Modell zu erhalten, muss die abstrakte Systemsicht auf die (elektrischen) DUT-Pins abgebildet werden [Ein99B]. So werden im Systemmodell digitale Signale z.B. als Integers oder als einzelne Steuerbits betrachtet, in der Realisierung aber über ein serielles Interface übertragen. Analoge Signale sind einfache Double Precision Real-Zahlen und müssen am Pin daher als Strom- oder Spannungswerte interpretiert werden. Im Gegensatz zur oben beschriebenen Verhaltensmodellierung ist die Verifikation dieser zusätzlichen Modellierung einfach [Ein99B].

Die hier beschriebene Methodik wurde im Förderprojekt VIRTUS an einem vier Kanal Analog Prozessor eines „Subscriber Line Interface and Codec Filter“ Schaltkreis demonstriert [Ein98].

Ein weiterer Ansatz besteht darin, die DUT-Modellierung nicht nur auf elektrischer Ebene, sondern auch auf Systemebene vorzunehmen [Ein99C]. Zusätzlich werden Pin-Modelle und Konverter, die zwischen abstrakter Systemdarstellung und elektrischer Darstellung platziert sind, verwendet. So können beispielsweise DSP-Algorithmen, Kontrollalgorithmen in VHDL sowie analoge Komponenten auf Transistorebene für die Modellierung eines Gesamtsystems angewendet werden. Diese Kombination führt zu einem detaillierten und sehr schnellen Modell. Die Umgebung für die hier vorgestellte Methode beinhaltet vier Tools die über ein IPC (Inter process communication) verbunden sind:

- IMAGE/ExChange
- Verilog
- COSSAP
- Leapfrog

Der Simulator COSSAP (system level simulator) übernimmt die Kontrollalgorithmen auf Systemebene (z.B. für DSP, Filter) sowie Leapfrog (VHDL Digital Simulator) die Steueralgorithmen auf der digitalen Seite (siehe Bild 2.6). Diese Kombination ermöglicht es ein schnelles DUT-Modell (DUT system level model) zu generieren.

Besteht der zu testende Schaltkreis hauptsächlich aus digitalen Blöcken, kann der IC mit einer digitalen Verhaltenssprache modelliert werden [Ron01]. Analoge Elemente werden mit Typ

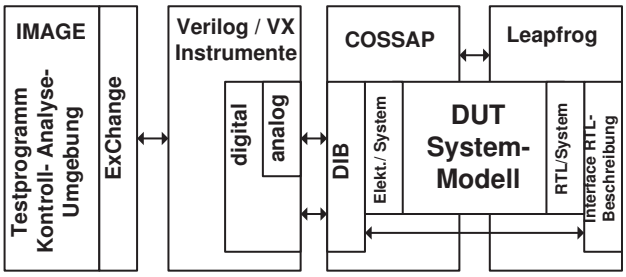

Abbildung 2.6: Virtual Test Equipment [Ein99C]

„REAL“ in VHDL beschrieben. Wenn es sich hierbei um einfache Komponenten, wie z.B. Verstärker handelt, kann das Wiederverwendungsprinzip zum Einsatz kommen. So erhält man ein sehr schnelles DUT-Verhaltensmodell, das mit der geeigneten VT-Umgebung sehr effektiv erstellt werden kann (z.B. mit Synthese [Rol97]).

Die Firma Dolphin beschäftigt sich ebenfalls mit dem virtuellen Test. In einer Veröffentlichung [Bab99] wird der Virtuelle Test von Komponenten mit der Verhaltenssprache VHDL-AMS vorgestellt. Im Speziellen wird ein DAC Generator (ADMIR) präsentiert, der auf dem Prinzip eines Sigma-Delta Modulator 1. Ordnung beruht. Dieser Generator erzeugt einen Digital-Analog-Konverter in VHDL-AMS für verschiedene Abstraktionsebenen, der im DUT-Modell an den Ein- und Ausgangspins verwendet werden kann. Hier wird unterschieden zwischen drei Simulationsmodellen mit unterschiedlichen Anforderungen und Feinheiten. Das algorithmische Modell wird durch eine mathematische Beschreibung von der Funktionalität der Komponente beschrieben. Als weiteres Modell wird ein „Performance measurement“-Modell vorgestellt. Dieses wird verwendet um das „nicht ideale“ Verhalten des Analogteils zu verifizieren. Der Digitalteil wird von dem algorithmischen Modell übernommen. Als letztes Modell wird das blockdetaillierte Modell präsentiert. Hierbei handelt es sich um ein Pin zu Pin kompatibles Modell mit nicht idealen Verhalten und einer digitalen Steuerung auf RTL-Ebene.
Der Simulator (SMASH), mit dem die Verhaltensmodelle simuliert werden, besteht aus einem Mixed-Signal und Mixed-Level Simulator, der VHDL-AMS als Beschreibungssprache unterstützt.

Bei der Firma Motorola [Fit03] hat sich für die Modellierung des DUTs die Verhaltenssprache MAST mit dem Simulator Saber etabliert. Die Vorteile für ein komplettes Verhaltensmodell liegen für die Firma nicht nur auf der Seite des virtuellen Tests, sondern auch bei der frühen Verfügbarkeit eines Modells für den Endkunden. Dieser kann somit bereits vor Lieferung des ersten Siliziums mit der Simulation seines Systems beginnen. Zur Generierung des DUT-Modells steht

eine umfangreiche Bibliothek zur Verfügung, die die Modellierungsarbeit erleichtern soll. Für digitale Komponenten wird der Simulator Verilog-XL zur Verfügung gestellt und erleichtert somit die Realisierung der Verhaltensmodelle.

Besonders geeignet für den VT sind Designs mit einer niedrigen Anzahl von Pins. Im Gegensatz dazu sind hochfrequente oder komplexe Designs nicht geeignet. Als Faustregel gilt, dass die Simulation eines einzigen Testschrittes nicht länger als 10 Minuten dauern sollte, um geeignet für den VT zu sein [Fit03].

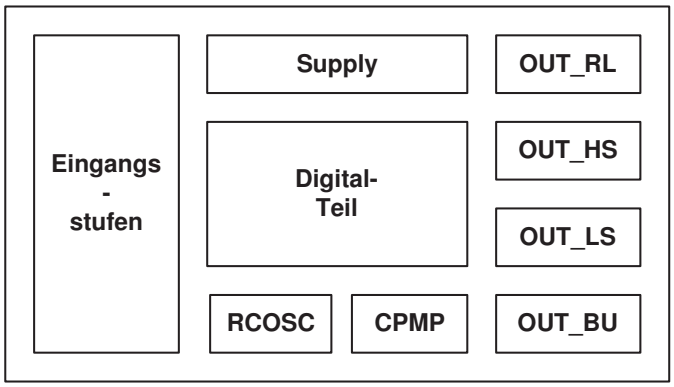

Abbildung 2.7: Blockschaltbild eines Mixed-Signal ICs

Ein konkretes Beispiel für die DUT-Modellierung wird im Förderprojekt VIRTUS geliefert [För99]. Hier dient als Demonstrator-IC ein Mixed-Signal ASIC aus dem Automobil-Bereich. In Bild 2.7 ist das Blockschaltbild ersichtlich, welches sowohl aus Netzlisten, als auch aus Verhaltensmodellen besteht. Der Digitalteil hat ca. 70 Module und 50 Grundgattertypen, die in Form

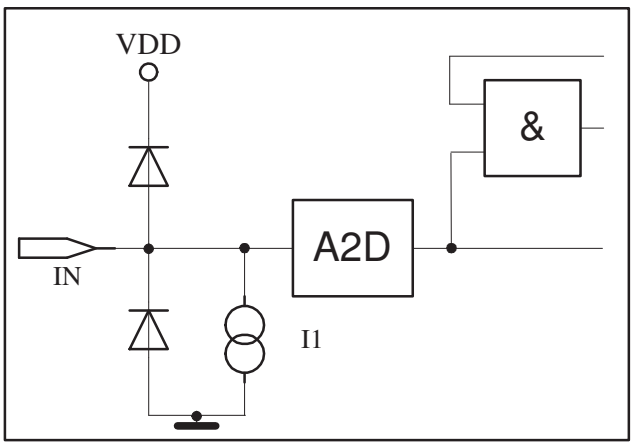

Abbildung 2.8: Modellierte Eingangsstufe des DUTs

einer Netzlistenbeschreibung im DUT vorliegen. Lediglich die Pins des ASICs mit allen analogen Eigenschaften, die beim Test erforderlich sind, wurden durch ein vereinfachtes Saber-Modell beschrieben. In Bild 2.8 und 2.9 werden zwei Beispiele daraus vorgestellt (Eingangsstufe und Ausgangsstufe).

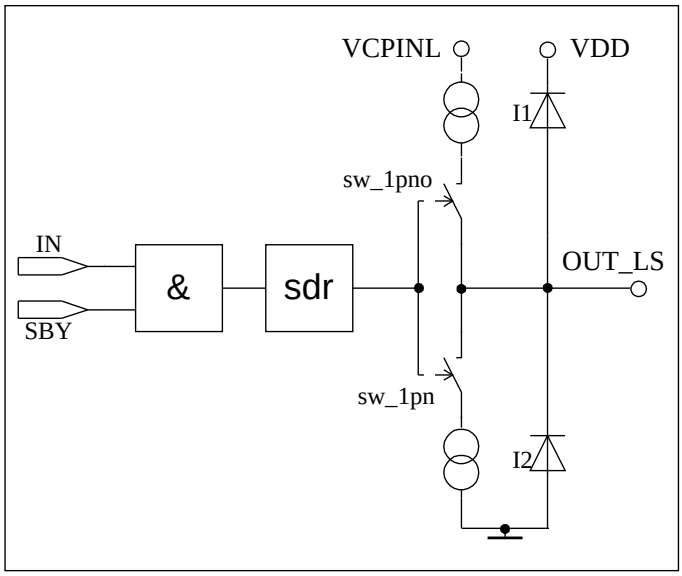

Abbildung 2.9: Modellierte Ausgangsstufe des DUTs

Bei beiden Stufen dienen Dioden als ESD-Schutzschaltung mit zusätzlichen Stromquellen für Pull-Down bzw. Pull-Up Effekte. Der A/D Konverter, der als Schnittstelle zwischen dem analogen Eingangspin und dem Digitalteil dient, ist mit einer Schaltschwelle und Delay-Zeit beschrieben worden. In der Ausgangsstufe steuert der Digitalteil zwei Schalter abwechselnd, so dass die Stromquellen entweder als „Push“ oder als „Pull“- Stufe benutzt werden.

Eine weitere Aktivität ist die Entwicklung eines zeit-effizienten Mixed-Signal Simulators für den virtuellen Test [Gras99]. Dieser Simulator (PESA) ist in der Lage, lange Pattern auf Mixed-Signal Ebene schneller als herkömmliche Simulatoren zu simulieren. Das Konzept beinhaltet Algorithmen für die Simulation analoger partitionierter Netzwerke als auch für die Simulation digitaler Blöcke. Zwei neue Syntax Anweisungen für eine Mixed-Signal Verhaltenssprache sind mit integriert worden. Diese Anweisungen beinhalten eine genaue Adressierung von Prozessen die durch globale Signale ausgelöst werden. Dadurch wird versucht, bei einem Zustandswechsel eines globalen Signals (z.B. clock), nur die Prozesse zu aktivieren in denen etwas passiert. Alle anderen sollen nicht aktiviert oder angesprochen werden. Dieses Prinzip verkürzt die Simulationszeit erheblich. Der Zeitgewinn beim Simulieren liegt um den Faktor 50.

2.5 MOSFET Level1-Modell

Im nachfolgenden Abschnitt soll das MOSFET Level1-Modell, welches im Laufe dieser Arbeit verwendet wird, kurz vorgestellt werden. Im Anhang D befindet sich eine detailliertere Beschreibung.
Das MOSFET Level1-Modell unterscheidet drei Arbeitsbereiche des Transistors, die sich durch die Spannungsdifferenzen zwischen Gate, Source und Drain definieren. Diese Arbeitsbereiche sind in Tabelle 2.1 aufgeführt.

Sperrbereich	$U_{GS} < U_{th}$
Linearer Bereich	$U_{GS} \geq U_{th} \wedge 0 \leq U_{DS} < U_{DS,sat}$
Sättigungsbereich	$U_{GS} \geq U_{th} \wedge U_{DS} \geq U_{DS,sat}$

Tabelle 2.1: Arbeitsbereiche im MOSFET Level1-Modell

Bild 2.10 zeigt das Ausgangskennlinienfeld eines NMOSFET-Transistors. Ist die Gate-Source-Spannung U_{GS} kleiner als die Schwellspannung U_{th}, kann sich kein leitender Kanal ausbilden. Damit befindet sich der Transistor im Sperrbereich, wo der Drainstrom $I_D \approx 0$ unabhängig von der Drain-Source-Spannung U_{DS} ist. Überschreitet U_{GS} die Schwellspannung U_{th}, bildet sich ein Kanal, in dem Strom fließen kann. Der Strom I_D ist für kleine U_{DS} näherungsweise proportional zu U_{DS}. Dieser Bereich wird daher als linearer Bereich bezeichnet. Wird U_{DS} über $U_{DS,sat} = U_{GS} - U_{th}$ erhöht, wird der Kanal auf der Drainseite abgeschnürt und I_D steigt nur noch sehr schwach an. Der Transistor befindet sich im Sättigungsbereich.

Der Drainstrom lässt sich unter der Annahme, dass eine ideale Ladungsverteilung im Kanal herrscht, näherungsweise mit Sah's Modell [Che99] berechnen. Das schwache Ansteigen von I_{DS} im Sättigungsbereich wird durch die Kanallängenmodulation verursacht. Sie lässt sich mit dem Shichman-Hodges-Modell [Che99] beschreiben. Damit lauten die Gleichungen für einen n-Kanal-MOSFET in den verschiedenen Arbeitsbereichen wie folgt:

$$I_D = \begin{cases} 0 & U_{GS} < U_{th} \\ \frac{K_n \cdot W}{L} U_{DS}(U_{GS} - U_{th} - \frac{U_{DS}}{2})(1 + \lambda U_{DS}) & U_{GS} \geq U_{th} \wedge 0 \leq U_{DS} < U_{DS,sat} \\ \frac{K_n \cdot W}{2L}(U_{GS} - U_{th})^2(1 + \lambda U_{DS}) & U_{GS} \geq U_{th} \wedge U_{DS} \geq U_{DS,sat} \end{cases} \tag{2.1}$$

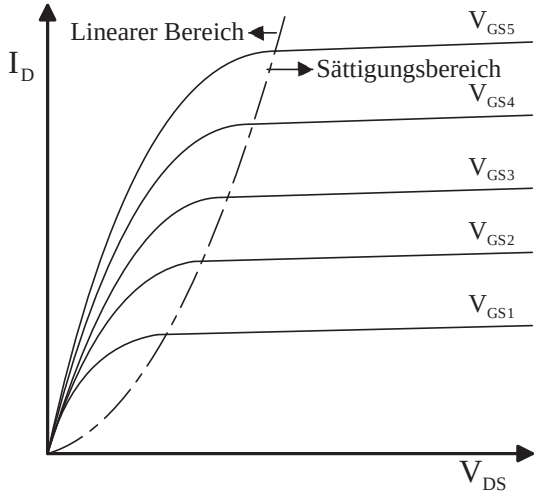

Abbildung 2.10: Kennlinie eines NMOSFET-Transistors

In Bild 2.11 ist das Blockschaltbild eines Level1-Modells dargestellt. Zusätzlich zu der Drainstromquelle I_D nach Gleichung 2.1 enthält es Bahnwiderstände, Kapazitäten und Bulk-Dioden.

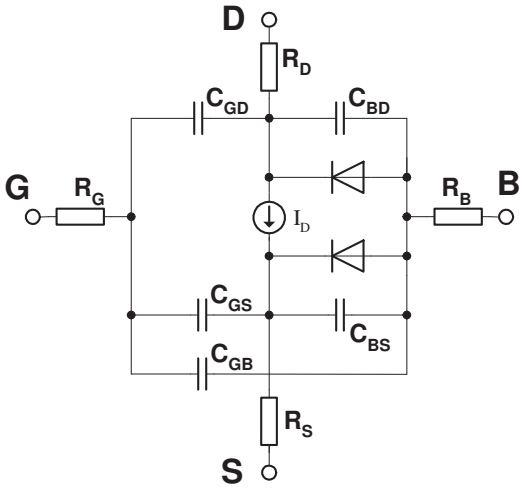

Abbildung 2.11: Blockschaltbild MOSFET Level1-Modell

Kurzkanal-Effekt:
Der in aktuellen Technologien auftretende Kurzkanal-Effekt führt zu einer Verschiebung der Schwellspannung bei Transistoren mit kurzen Kanallängen [Che99]. Dieser Effekt findet im ursprünglichen MOSFET Level1-Modell keine Berücksichtigung. Für das in Abschnitt 4.3 vorge-

stellte Modell wurde der Effekt durch den in [Che99] vorgeschlagenen Faktor

$$\ldots \cdot \left(1+\lambda \left(\frac{1+\theta \cdot L_{eff}}{\theta \cdot L_{eff}}\right) \cdot U_{DS}\right) \tag{2.2}$$

in Gleichung 2.1 hinzugefügt.

2.6 Die Hardwarebeschreibungssprache Verilog-AMS

In den 90er Jahren wurde die Forderung nach einer standardisierten Mixed-Signal Sprache immer größer. Einer der ersten Ansätze einer Mixed-Signal Sprache war MAST, die aber den Nachteil hatte nicht standardisiert gewesen zu sein. Die meisten EDA Hersteller versuchten dieses Problem mit Simulatorkopplungen zu entschärfen, was aber einen Performanceverlust zur Folge hatte. Hier sind beispielsweise Continuum von Mentor oder SpectreVerilog von Cadence zu nennen, die eine Kopplung zwischen analogen und digitalen Simulatoren darstellen. Mit VHDL-AMS [Chr99] wurde 1999 eine Standardisierung vorgenommen die in „IEEE Standard 1076.1-1999“ beschrieben ist. Verilog-AMS wurde 2001 (IEEE 1364-1995 Verilog HDL) standardisiert mit dem Ziel Mixed-Signal bzw. Mixed-Domain Systeme in einer Sprache beschreiben zu können. Verilog-AMS [Tri97], [Mil00] ist die Zusammenführung und Erweiterung der Sprachen Verilog-D und von Verilog-A [Fit03]. Diese drei Sprachen bilden derzeit die Verilog Familie (Bild 2.12) [Kun04].

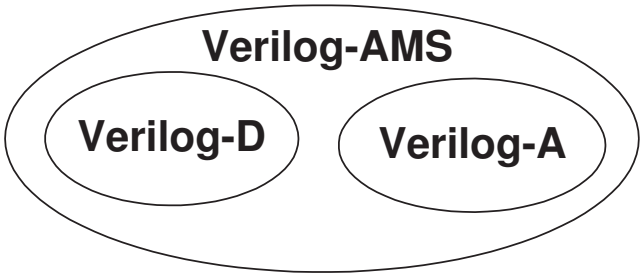

Abbildung 2.12: Die Beziehung zwischen Verilog-AMS, Verilog-A und Verilog-D

Wie bereits erwähnt, beschreibt Verilog-D die digitalen und Verilog-A die analogen Komponenten. Verilog-AMS kombiniert diese zwei Sprachen und fügt zusätzliche Erweiterungen hinzu, um die Beschreibung von Mixed-Signal Komponenten zu ermöglichen.

Digitale Signale sind für eine bestimmte Zeitperiode konstant und nehmen dann abrupt einen neuen Wert an (=ereignisgesteuerte Simulation [Zei84]). Die Sprache Verilog-D wurde entwickelt, um logische Signalzustände zu verarbeiten und Systeme zu gestalten, die diese generieren. Analoge Signale variieren kontinuierlich, d.h. der Wert eines Signals kann zu jedem Zeitpunkt

jeden beliebigen Wert innerhalb eines bestimmten Bereiches annehmen (=zeitkontinuierliche Simulation [Zei84]).

Seit Verilog-AMS die Sprachen Verilog-D und Verilog-A verbindet, ist es auch möglich, mit Systemen umzugehen, die digitale und zeitkontinuierliche analoge Signale beinhalten. Zusätzlich ermöglicht es die Unterstützung von Systemen, die analoge Einzelereignisse behandeln.

Experten der Verhaltensmodellierung nehmen an, dass Verilog-AMS einen großen Einfluss auf das Design von Mixed-Signal Systemen haben wird, weil es eine einzige Sprache und einen einzigen Simulator vereint, der zwischen Analog- und Digitalentwicklern und zwischen Baustein- und Systementwicklern aufgeteilt ist [Kun04]. Durch Verilog-AMS ist es nun möglich, Verhaltensmodelle für Mixed-Signal Blöcke zu schreiben. Desweiteren werden die ereignisgesteuerten Fähigkeiten auf die Analogsimulation übertragen und es wird ermöglicht, analoge Modelle zu schreiben, die die Geschwindigkeit und Eigenschaften von Digitalsimulation geerbt haben.

Dies ist sehr wichtig, da viele der analogen und Mixed-Signal Modellen, die in Simulationen auf höherer Ebene genutzt werden, normalerweise so geschrieben sind, dass sie ereignisgesteuerte Konstrukte verwenden. Bausteine wie zum Beispiel Analog-Digital-Wandler, Digital-Analog-Wandler oder PLLs (Phasenregelschleife) sind leichte und sehr effiziente Modelle, die analoge ereignisgesteuerte Merkmale der AMS Sprache verwenden.

2.6.1 Aufbau eines Verilog-D Modells

Das grundlegende Strukturierungskonzept in Verilog ist das Modul. Die Definition eines Moduls legt die Ein- und Ausgänge fest und beinhaltet eine Beschreibung des Verhaltens des Moduls. Es ist syntaktisch wie in Bild 2.13 aufgebaut.

Modulbeschreibungen können nicht geschachtelt werden, d.h. es gibt keine Moduldefinitionen innerhalb von Modulen. Eine Hierarchie innerhalb eines Designs wird erreicht, indem innerhalb eines Moduls weitere Module instanziiert werden können. Zusätzlich können dem Modell über Bibliotheken (Libraries) und Packages, häufig benötigte Datentypen, Objekte und Funktionen zugänglich gemacht werden.

Als Beispiel für den Aufbau eines Moduls ist in Bild 2.14 ein Ohmscher Widerstand dargestellt.

```
module resistor (p,n)        // Modulname, Pinauflistung
inout p,n;                   // Pinbeschreibung
electrical p,n;              // Signaltyp-Beschreibung
parameter real r = 100.0;    // Parameter-Definition
analog begin
 V(p,n) <+ I(p,n)*r;         // Beschreibung des Verhaltens
end
endmodule
```

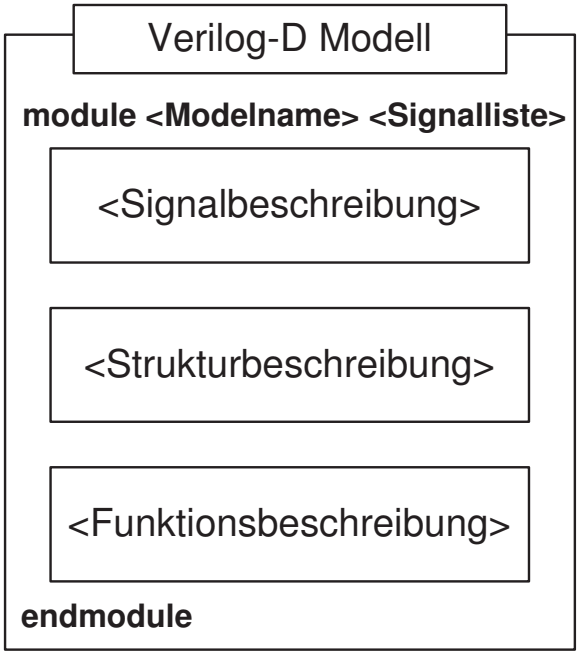

Abbildung 2.13: Struktur eines Verilog-D Modells

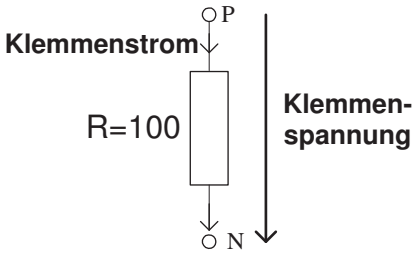

Abbildung 2.14: Verilog-A Modell am Beispiel eines Widerstandes

Die Moduldefinition wird von den Schlüsselwörtern „module“ und „endmodule“ eingerahmt. Somit wird zuerst der Modulname festgelegt, dem alle Ein- und/oder Ausgänge in geschweifter Klammer folgen. Der beschriebene Widerstand hat somit zwei Pins (p und n). Anschließend wird festgelegt, um welche Art von Klemme es sich handelt und um welchen Signaltyp. Da die beiden Pins sowohl als Ein- wie auch als Ausgänge verwendet werden können, werden sie durch „inout“ als bidirektionale Pins gekennzeichnet. Bei der Parameterdefinition wird der Wert des Widerstandes bestimmt, der in diesem Beispiel 100 Ohm beträgt. Nach dem Schlüsselwort „analog“ folgt die eigentliche analoge Verhaltensbeschreibung. Sie muss durch die Schlüsselwörter „begin“ und „end“ eingerahmt werden. Der reale Widerstand lässt sich nach dem Ohmschen Gesetz wie folgt

berechnen:

$$U = I * R \tag{2.3}$$

Zunächst wird die Spannung über den Klemmen „p“ und „n“ durch die Schreibweise „V(p, n)“ bestimmt und anschließend der Strom „I“ zwischen den Klemmen „p“ und „n“ aus dem Quotient der Spannung und Widerstand errechnet. Damit ist die grundlegende Struktur eines Verilog-A Codes erläutert, die darüber hinaus mit der Strukturierung des digitalen Verilog identisch ist.

2.6.2 Analoge Operatoren in Verilog-A/AMS

In Verilog-AMS gibt es eine Reihe Operatoren, die es ermöglichen analoges Verhalten nachzubilden. Diese Operatoren stehen ebenfalls in der analogen HDL Verilog-A zur Verfügung.

Alle analoge Operatoren sind im [LRM04] ausführlich beschrieben.

Analoger Operator	**Beschreibung**
ddt	Differenzial
ddx	Differenzial
idt	Zeit-Integration
idtmod	Winkel-Integration
transistion	Realisierung von Signalübergängen
slew	Anstiegsfunktion
absdelay	Verzögerungsfunktion
last_crossing	Ausgabe der Simulationszeit
above	Schwellwertabfrage
cross	Schwellwertabfrage
limexp	limitierte Exp.-Funktion
laplace_zp	Laplacefunktion
laplace_zd	Laplacefunktion
laplace_np	Laplacefunktion
zi_zp	Z-Übertragungsfunktion
zi_zd	Z-Übertragungsfunktion
zi_np	Z-Übertragungsfunktion
timer	Zähler

Tabelle 2.2: Analoge Operatoren in Verilog-AMS

Beschreibung des Transition Filters:
Eine sehr häufig verwendeter Analog-Operator ist der „Transistion Filter“, der es ermöglicht, Verzögerungen und Signalübergänge nachzuahmen. Nach dem eigentlichen „Wert“ des Signals können Verzögerungs- (dt), Anstiegs- (tr) und Abfallzeit (tf) angegeben werden. Aus einem idealen Signalverlauf wie er im Bild 2.15 zu sehen ist, wird ein reeller Verlauf mit den entsprechenden Verzögerungszeiten.

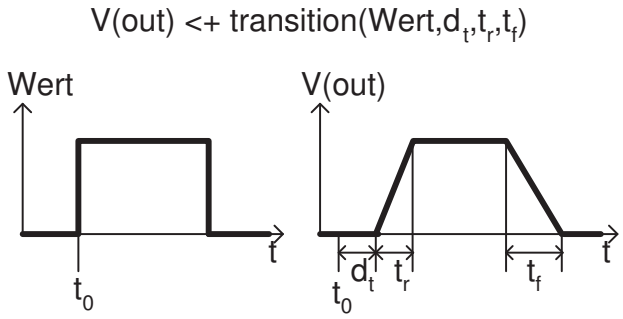

Abbildung 2.15: Transition Filter

2.6.3 Kommandos für Mixed-Signal Operationen

In Verilog-AMS gibt es getrennte Abschnitte für analoge und digitale Operationen. Die analogen Anweisungen stehen innerhalb des Analogblockes, der mit „analog begin“ anfängt und mit „end“ endet. Die digitalen Operationen stehen außerhalb des Analogblockes, meist beginnend mit einer „@always()“, oder „@initial“ Anweisung, so wie es die reine Verilog Sprache erlaubt. Das Generieren eines Signales, sei es nun eine Spannung für den Analogteil oder ein logisches Signal für den Digitalteil, kann nur innerhalb des jeweiligen Blockes geschehen. Eine analoge Spannung kann also nur im analogen Block und ein digitales Signal nur in einem digitalen Block, einem Ausgangspin zugewiesen werden.

Was sehr wohl möglich ist, ist das Beobachten einer analogen Größe im Digitalblock, und das Erkennen eines digitalen Zustandes im Analogblock. Diese sind die Kommandos und die Funktionalität die speziell für die Mixed-Signal Sprache Verilog-AMS implementiert wurden. Hierbei handelt es sich zur Überwachung von analogen Signalen im Digitalblock um folgende Befehle:

- above

- cross

Für die Überwachung der digitalen Signale im Analogblock können die nachfolgenden Anweisungen verwendet werden:

- if-else
- case
- @(Operator) (z.B. above, cross, posedge, negedge)

Beispiel case:
Im nachfolgenden Listing ist ein Beispiel für einen Digital-Analog- Wandler dargestellt, in dem die „case“ Funktion zum Detektieren des digitalen Eingangszustands verwendet wird.

```
module dtoa(d, a);
input d;
logic d;
output a;
electrical a;
real var;
analog begin
  case (d)
     1'b1:var = 5;       // aktiv
     1'bx:var = var;     // den Wert halten
     1'b0:var = 0;       // inaktiv
     1'bz:var = 2.5;     // hochohmig
  endcase
 V(a) <+ var;            // analoge Anweisung
end
endmodule
```

2.6.4 Beispiele für analoge Komponenten

In diesem Kapitel werden zunächst einfache Beispiele beschrieben, um die Einführung in die Sprache Verilog-AMS zu vertiefen. Es werden analoge und digitale Module erläutert, die mit geringen Veränderungen auch eigenständig in Verilog-A bzw. in Verilog-D beschrieben werden könnten. Am Ende dieses Kapitels werden dann Beispiele aufgezeigt, die beide Sprachen vereinen.

Kondensator:
Im folgenden Beispiel wird ein Kondensator beschrieben.

```
module capacitor (p,n)          // Modulname, Pinauflistung
inout p,n;                      // Pinbeschreibung
electrical p,n;                 // Signaltyp-Beschreibung
```

```
parameter real c = 1.0  ;        // Parameter-Definition
branch b_cap(p,n);

analog begin
  I(b_cap) <+ c* ddt(V(b_cap));  // Verhaltensbeschreibung
end
endmodule
```

Zu Beginn wird das Modul deklariert und die Ports beschrieben. Mit dem Schlüsselwort „electrical“ werden die Ports als analoge Signale festgelegt. Über die eingeladene Library und das Schlüsselwort ist nun bekannt, dass es sich bei der Domäne „electrical“ um die Potentialgröße Spannung und die Flussgröße Strom handelt. Desweiteren sind nun die physikalischen Eigenschaften definiert. Analoge Potentiale und Ströme können nur in einem Analogblock dem Knoten oder Zweig zugeteilt werden (Schreibmodus). Sie können sowohl in digitalen als auch analogen Verfahrensblöcken gelesen werden (Lesemodus). Im vorherigen Beispiel erfolgte die Zuweisung an beide Klemmen. In diesem Beispiel wird eine weitere Möglichkeit aufgezeigt. Mit dem Schlüsselwort „branch“ werden die Pins zu einem Zweig zusammengefasst und können nun im Analogblock direkt dem Stromzweig zugewiesen werden.

Der Parameter „c“ kann in diesem Beispiel einen Wert zwischen Null und unendlich annehmen. Er kann im Symbol festgelegt werden oder nimmt den Defaultwert Null an. In der analogen Verhaltensbeschreibung wird nun der Kondensator beschrieben. Durch die Verwendung der nachfolgenden Gleichung wird das Verhalten des Kondensators nachgebildet:

$$I = C * \frac{d}{dt}(U) \tag{2.4}$$

In Verilog bedeutet „ddt“ die zeitliche Ableitung der nachfolgenden Zweigspannung. Die Spannung über dem Zweig wird differenziert, mit dem Parameter „c“ multipliziert und somit der Strom durch den Zweig bestimmt. Mit Hilfe dieser Implizierungen wird das Verhalten eines Elements beschrieben.

Spannungsgesteuerte Spannungsquelle:
Im nachfolgenden Beispiel wird eine spannungsgesteuerte Spannungsquelle beschrieben. Zwischen den Eingangspins „pc“ und „nc“ wird eine Spannung gemessen und an den beiden Ausgangspins „p“ und „n“ mit dem Verstärkungsfaktor „g“ ausgegeben.

$$U_{p,n} = g * U_{pc,nc} \tag{2.5}$$

```
module vcvs (p,n,pc,nc)      // Modulname, Pinauflistung
inout p,n,pc,nc;             // Pinbeschreibung
```

```
electrical p,n,pc,nc;        // Signaltyp-Beschreibung
parameter real g = 10.0  ;   // Parameter-Definition

analog begin
 V(p,n) <+ g* V(pc,nc));     // Verhaltensbeschreibung
end
endmodule
```

2.6.5 Beispiele für Mixed-Signal Komponenten

Die zuvor vorgestellten Beispiele verdeutlichen noch nicht die Vorteile einer Mixed-Signal HDL. In den nächsten Beispielen sollen diese Vorteile verdeutlicht werden.

Digital-Analog-Wandler:
Ein Digital-Analog-Umsetzer hat die Aufgabe ein digitales Signal in einen Spannungswert (analoges Signal) umzuwandeln. Da bei einem D/A-Wandler ein digitales Eingangssignal vorliegt und ein analoges Ausgangssignal erzeugt werden soll, müssen auch die Ports unterschiedliche Signaltypen aufweisen („logic“ und „electrical“). Mit den Parametern „vhi“ und „vlo“ werden die analogen Ausgangspannungen festgelegt.
Im nachfolgenden Listing ist das Beispiel eines Verilog-AMS beschriebenen D/A-Konverters zu sehen.

```
module imple_dtoa (inp,outp)  // Modulname, Pinauflistung
input inp;                    // Pinbeschreibung
output outp;
electrical outp;              // Signaltyp-Beschreibung
logic inp;
parameter real vhi = 5.0  ;   // Parameter-Definition
parameter real vlo = 0.0  ;   // Parameter-Definition

real vout;
analog begin
@(initial_step) vout = 0;     // Initialisierung
 if (inp == 1)                // wenn in "high" dann
   vout=vhi;                  // vout = vhi
 else                         // sonst
   vout=vlo;                  // vout = vlo

 V(outp) <+ transition(vout); // Verhaltensbeschreibung
```

```
end
endmodule
```

Im Analog-Block wird nun das digitale Eingangssignal „inp“ abgefragt. Stimmt dies mit dem logischen Wert „1“ überein, dann wird dem internen Signal „vout“ der Wert von „vhi“ („5“) übergeben. Wenn dies nicht übereinstimmt, wird der Wert von „vlo“ („0“) übergeben. Der Wert des internen Signals „vout“ wiederum wird dem Port „out“ über den Analog-Operator „transition“ zugewiesen.

Analog-Digital-Wandler:
Bei diesem A/D-Wandler wird nun nicht zu jedem Zeitpunkt das Signal „inp“ überprüft, sondern nur bei Signaländerungen. In zwei Always-Blöcken werden die Schaltschwellen über den „cross“ Befehl abgefragt. Im ersten Block wird die steigende Taktflanke („1“) auf den Wert 2,5 Volt abgefragt. Wird dieses Signal erkannt, wird der Ausgang auf logisch („1“) gesetzt. Im zweiten Block erfolgt dieselbe Abfrage auf die fallende Taktflanke („-1“), die ein Signal logisch („0“) zur Folge hat. Bevor keine der beiden genannten Signaländerungen stattgefunden hat, befindet sich das Ausgangssignal in einem undefinierten Zustand. Zur Abhilfe wird im Initial-Block das Signal auf einen definierten Wert gebracht. Der Initial-Block wird nur einmal zum Programmbeginn durchlaufen.

```
module simple_atod(inp,cm);
input inp;
output cm;
reg cm;
electrical inp;
logic cm;

always@(cross(V(inp)-2.5, 1))  //wenn inp > 2.5V
   cm = 1;                     //dann cm = 1
always@(cross(V(inp)-2.5, -1)) //wenn inp < 2.5V
   cm = 0;                     //dann cm = 0
endmodule
```

Der Vorteil von Verilog-AMS ist das Verwenden von analogen und digitalen Blöcken innerhalb eines Modells. So können beispielsweise im Digitalblock analoge Signale durch die Anweisungen above und cross überwacht werden. In Gegensatz dazu müsste z.B. bei Verilog-A die Überwachung im analogen Bereich stattfinden, was einen erheblichen Performanceverlust zur Folge hätte.

Kapitel 3

Modellierungsmethodik

Heutzutage werden nicht nur bei der Schaltungsentwicklung sondern auch in anderen Bereichen wie bei der Test-, ESD- oder Applikationsentwicklung Verhaltensmodelle benötigt. Ebenso werden Modelle der Schaltkreise frühzeitig vom Kunden gefordert („Kundenmodelle“). Auch innerhalb der Schaltungsentwicklung gibt es unterschiedliche Anwendungsbereiche, die Verhaltensmodelle mit unterschiedlichen Eigenschaften, und unterschiedlicher Komplexität benötigen. Neben der Systementwicklung und der eigentlichen Entwicklung der Schaltung werden auch bei der abschließenden Verifikation der Schaltung Verhaltensmodelle in verschiedenen Detaillierungsgraden benötigt. Der zunehmende Einsatz von Verhaltensmodellen im Laufe der Produktentwicklung, die unterschiedliche Anforderungen beinhalten müssen, verursacht ein nicht zu vernachlässigten Modellierungsaufwand, den es zu minimieren gilt.

Der Einsatz von virtuellen Testmethoden ist in der Industrie häufig mit der Frage an die Wirtschaftlichkeit verbunden. Ziel ist es das Verhaltensmodell für den DUT mit möglichst wenig Zeitaufwand zu erstellen und die hohen Simulationszeiten, die bei VT-Simulationen entstehen können, zu reduzieren. Um diesen Forderungen nachzukommen, müssen Verfahren eingesetzt werden, die den Modellierungsaufwand minimieren und trotzdem den speziellen Anforderungen an ein DUT-Modell für den VT gerecht werden.

Der Einsatz von Verhaltensmodellen ermöglicht es, bei der Entwicklung einer Applikation frühzeitig durch Simulation Probleme zu lokalisieren. Ebenso wird bei der Entwicklung von Software z.B. „Firmware“ ein Verhaltensmodell für eine Hardware/Software Cosimulation benötigt. Hier werden wie bei der Schaltungsentwicklung Modelle je nach Einsatzgebiet mit verschiedenen Eigenschaften benötigt.

Ein weiterer Bereich für Verhaltensmodelle stellt die EMV (Elektromagnetische Verträglichkeit) Simulation dar. Besonders in der Automobilindustrie sind diese Untersuchungen, meist auf Systemebene (z.B. CAN-Bus im Fahrzeug), ein wichtiges Thema. Hierfür werden Verhaltens-

modelle mit speziellen Eigenschaften benötigt.

In dem nachfolgenden Kapitel wird eine bereichsübergreifende Modellierungsmethodik vorgestellt. Sie hat das Ziel, die erstellten Verhaltensmodelle variabel und für alle Anwendungsbereiche einsetzen zu können.

3.1 Stand der Technik

In der Schaltungsentwicklung haben sich Entwurfsstrategien wie „Top-Down-Designmethodik“ oder „Meet-In-The-Middle“ (siehe Kapitel 2.2), etabliert. Bei diesen Entwurfsmethoden werden Verhaltensmodelle in verschiedenen Abstraktionsebenen benötigt (siehe Kapitel 2.1). Das Erstellen der Verhaltensmodelle kann mit verschiedenen Methoden stattfinden (siehe Kapitel 2.3). Die Modelleigenschaften bzw. Anforderungen müssen hierbei vor der Modellierung definiert und den verschiedenen Modellrepräsentationen zugewiesen werden. Dies bedeutet, dass die Verhaltensmodelle auf den niedrigsten Abstraktionsebenen meist alle Modelleigenschaften beinhalten. Diese Modelle sind sehr genau, aber meist auch rechenintensiv.

Bei der Erstellung der Modelle werden in den meisten Fällen typische Modellparameter, wie beispielsweise die Verstärkung von einem OpAmp oder die Grenzfrequenz von einem Tiefpass variabel als Parameter übergeben. Eine gezieltes Aktivieren oder Deaktivieren von Modelleigenschaften (z.B. das dynamische Verhalten bzw. die Übertragungsfunktion eines OpAmps) ist hierbei nicht vorgesehen.

Die Austauschbarkeit unter den Modellen der unterschiedlichen Abstraktionsebenen stellt ein weiteres Problem dar (z.B. im Top-Down-Entwurf die unterschiedliche Realisierung der Peripherie der Modelle). Nicht nur die Anzahl der Pins, sondern auch der Übergang von „nichtkonservativen“ zu „konservativen“ Klemmen ist problematisch. Erste Lösungsansätze gibt es in [Jor01].

In der Praxis werden die Modelle im Laufe der Verwendung immer wieder um Modelleigenschaften, die bei der Erstellung keine Berücksichtigung fanden, erweitert. Dies führt zu einem nicht zu vernachlässigenden Mehraufwand. Werden z.B. „Black-Box“- Modellierungsmethoden, die das Verhalten mit Approximation und Interpolation nachbilden (siehe [Dam97], [Ant99]) angewandt, muss die Modellerstellung wiederholt werden. Hierbei müssen zusätzlich die Stimuli, die das gewünschte Verhalten sichtbar machen, neu definiert werden.

Verhaltensmodelle werden nicht nur im Bereich der Schaltungsentwicklung sondern beispielsweise auch beim virtuellen Test eingesetzt. In den meisten Fällen können die Modelle, die bei der Schaltungsentwicklung verwendet werden, nicht beim VT eingesetzt werden. Diese müssen über alle Abstraktionsebenen erweitert bzw. angepasst werden (siehe Kapitel 2.4).

Erste Ansätze für EMV-Modelle sind mit „IBIS“-Modellierungsverfahren verbreitet [Wan99]. Bei Modellen für Applikationen werden ähnliche Ansätze wie bei der Realisierung von Loadboard-Modellen [Leh07] für den virtuellen Test eingesetzt [Ant02].

Weitere Ansätze in der Literatur sind begleitend zu den Grundlagen der Arbeit in Kapitel 2 beschrieben.

3.2 Modelleigenschaften

Für eine Modellierungsmethodik, die den gesamten Entwurfsablauf berücksichtigen soll, müssen zu Beginn der Modellerstellung Modelleigenschaften und somit das Modellverhalten aus allen Bereichen der Produktentwicklung festgelegt werden. Die Ermittlung dieser Eigenschaften stellt die Basis für die hier vorgestellte Methodik dar. Im nachfolgenden Kapitel sollen diese Eigenschaften definiert und klassifiziert werden, um diese bei der Generierung der Verhaltensmodelle zu nutzen.

Die Modelleigenschaften e_i sind in der Menge E definiert durch:

$$E = \{e|e_1, ..., e_n\} \tag{3.1}$$

Hierbei stellen $e_1, ..., e_n$ die Modelleigenschaften und somit die so genannten „Urelemente“ der Eigenschaftsmenge E dar. Bei den heutigen Modellierungsansätzen werden je nach Anwendung gezielte Eigenschaften ausgewählt, die durch folgende Teilmenge beschrieben werden kann:

$$R \subset E \tag{3.2}$$

$$r_1, ..., r_m \in E \tag{3.3}$$

Problematisch ist hierbei die große Vielfalt der möglichen Teilmengen und somit der Eigenschaften, die in Verhaltensmodelle implementiert werden können. Jede Teilmenge stellt hierbei die Anforderungen für ein zu realisierendes Modell dar (siehe Bild 3.1). Dies bedeutet ein nicht zu vernachlässigenden Modellierungsaufwand.
In dem nachfolgenden Abschnitt soll eine Methode vorgestellt werden, bei der der komplette Eigenschaftsraum bei der Erstellung von Verhaltensmodellen berücksichtigt werden kann.

In der Menge E, die alle Modelleigenschaften beinhaltet, gibt es „ähnliche“ Eigenschaften, d.h. identische Modelleigenschaften in verschiedenen Detaillierungsgrade (siehe Bild 3.2). Dieses erschwert die Betrachtung des Eigenschaftsraums für die eindeutige Auswahl von Modelleigenschaften.

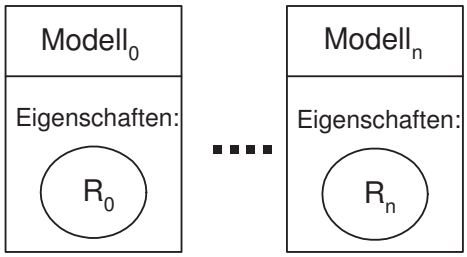

Abbildung 3.1: n-Anzahl von Modellen mit verschiedenen Modelleigenschaften

In Bild 3.2 stellt $\widetilde{e_1}$ eine Modelleigenschaft auf hoher Abstraktionsebene dar. Die gleiche Eigenschaft auf einer tieferen Abstraktionsebene beinhaltet jedoch eine Untermenge mit den Eigenschaften $\widetilde{e_I}$, $\widetilde{e_{II}}$ und $\widetilde{e_{III}}$. Diese Modelleigenschaften können dann wieder auf einer noch tieferen Abstraktionsebene durch eine Untermenge von E beschrieben werden z.B. $\widetilde{e_I}$ durch $\widetilde{e_a}$, $\widetilde{e_b}$ und $\widetilde{e_c}$. Die Modelleigenschaften $\widehat{e_2}$ und $\widehat{e_{IV}}$ haben hierbei keine „ähnlichen“ Modelleigenschaften auf einer niedrigeren Abstraktionsebene. Hierbei soll für diese Arbeit folgende Notation gelten:

Definition 3.1
$\widetilde{e_x}$ = „ähnliche“ Modelleigenschaft auf einer tieferen Abstraktionsebene vorhanden
$\widehat{e_y}$ = keine „ähnliche“ Modelleigenschaft auf einer tieferen Abstraktionsebene vorhanden

wobei gelten muss:

$$\widehat{E} = \{\widehat{e}|\widehat{e_1}, ..., \widehat{e_y}\} \tag{3.4}$$

$$\widetilde{E} = \{\widetilde{e}|\widetilde{e_1}, ..., \widetilde{e_x}\} \tag{3.5}$$

$$E = \widehat{E} \cup \widetilde{E} \tag{3.6}$$

Es sollen an dieser Stelle drei Mengen für die Modelleigenschaften definiert werden, mit dem Ziel, keine „ähnliche“ Eigenschaften innerhalb einer Menge zu erhalten.

Definition 3.2
Es werden die Mengen L_1, L_2 und L_3 definiert durch:

$$L_1 = \widetilde{L_1} \cup \widehat{L_1} \tag{3.7}$$

$$L_2 = \widetilde{L_2} \cup \widehat{L_2} \tag{3.8}$$

$$L_3 = \widetilde{L_3} \cup \widehat{L_3} \tag{3.9}$$

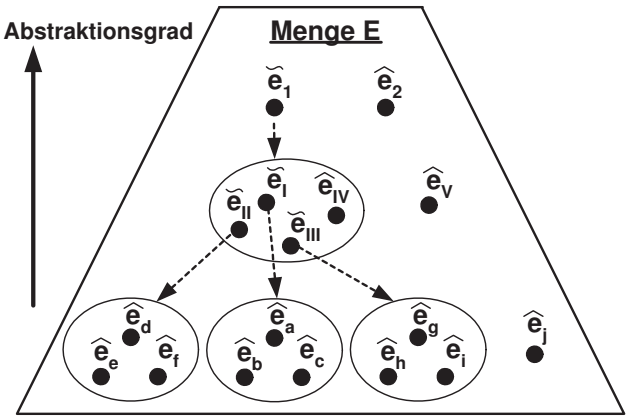

Abbildung 3.2: Identische Modelleigenschaften in verschiedenen Abstraktionsebenen

Hierbei wird $\widehat{L_x}$ und $\widetilde{L_x}$ wie folgt definiert:

$$\widehat{L_1} = \left\{\widehat{l_1}|\widehat{l_{11}},...,\widehat{l_{1m}}\right\} \tag{3.10}$$

$$\widetilde{L_1} = \left\{\widetilde{l_1}|\widetilde{l_{11}},...,\widetilde{l_{1p}}\right\} \tag{3.11}$$

$$\widehat{L_2} = \left\{\widehat{l_2}|\widehat{l_{21}},...,\widehat{l_{2n}}\right\} \tag{3.12}$$

$$\widetilde{L_2} = \left\{\widetilde{l_2}|\widetilde{l_{21}},...,\widetilde{l_{2q}}\right\} \tag{3.13}$$

$$\widehat{L_3} = \left\{\widehat{l_3}|\widehat{l_{31}},...,\widehat{l_{3o}}\right\} \tag{3.14}$$

$$\widetilde{L_3} = \left\{\widetilde{l_3}|\widetilde{l_{31}},...,\widetilde{l_{3r}}\right\} \tag{3.15}$$

wobei immer noch gelten muss:

$$\widehat{l_{11}},...,\widehat{l_{1m}} \in E \tag{3.16}$$

$$\widetilde{l_{11}},...,\widetilde{l_{1p}} \in E \tag{3.17}$$

$$\widehat{l_{21}},...,\widehat{l_{2n}} \in E \tag{3.18}$$

$$\widetilde{l_{21}},...,\widetilde{l_{2q}} \in E \tag{3.19}$$

$$\widehat{l_{31}},...,\widehat{l_{3o}} \in E \tag{3.20}$$

$$\widetilde{l_{31}},...,\widetilde{l_{3r}} \in E \tag{3.21}$$

Es soll weiter gelten:

$$\widehat{l_{11}},...,\widehat{l_{1m}} \notin L_2 \cup L_3 \tag{3.22}$$
$$\widetilde{l_{11}},...,\widetilde{l_{1p}} \notin L_2 \cup L_3 \tag{3.23}$$
$$\widehat{l_{21}},...,\widehat{l_{2n}} \notin L_1 \cup L_3 \tag{3.24}$$
$$\widetilde{l_{21}},...,\widetilde{l_{2q}} \notin L_1 \cup L_3 \tag{3.25}$$
$$\widehat{l_{31}},...,\widehat{l_{3o}} \notin L_1 \cup L_2 \tag{3.26}$$
$$\widetilde{l_{31}},...,\widetilde{l_{3r}} \notin L_1 \cup L_2 \tag{3.27}$$
$$\tag{3.28}$$

Die Gesamtmenge E ist somit die Summe der Untermengen L_1, L_2 und L_3

$$E = L_1 \cup L_2 \cup L_3 \tag{3.29}$$

ebenso gilt:

$$E = \widetilde{L_1} \cup \widehat{L_1} \cup \widetilde{L_2} \cup \widehat{L_2} \cup \widetilde{L_3} \cup \widehat{L_3} \tag{3.30}$$

mit

$$L_1 \subset E \tag{3.31}$$
$$L_2 \subset E \tag{3.32}$$
$$L_3 \subset E \tag{3.33}$$

In Bild 3.3 ist das Euler-Vennsche Diagramm [Bor87] der oben beschriebenen Definition abgebildet.

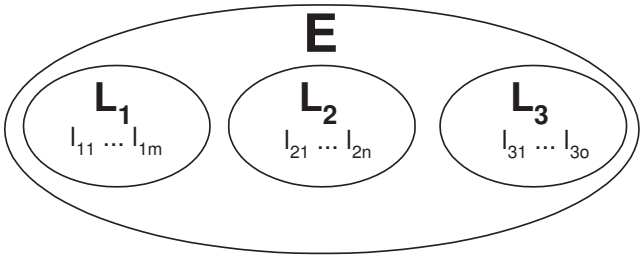

Abbildung 3.3: Euler-Vennsche Diagramm der Mengen L_1, L_2 und L_3

Bei der Modellerstellung sollen alle **M**odell-**E**igenschaften (ME) aus L_1, L_2 und L_3 mit einfließen. Die Modelle sollen es ermöglichen, Eigenschaften nach folgender Regel auszuwählen:

$$m_{ME,n} \subseteq L_1 \cup L_2 \cup L_3$$

Weiter können auch Teilmengen G_1, G_2 oder G_3 aus den Mengen L_1, L_2 oder L_3 eine Modelleigenschaft darstellen.

$$G_1 \subset L_1 \text{ mit } ((g_{11}...g_{1m}) \in L_1) \tag{3.34}$$

$$G_2 \subset L_2 \text{ mit } ((g_{21}...g_{2n}) \in L_2) \tag{3.35}$$

$$G_3 \subset L_3 \text{ mit } ((g_{31}...g_{3o}) \in L_3) \tag{3.36}$$

In Bild 3.4 ist zur Verdeutlichung das Euler-Vennsche Diagramm der oben beschriebenen Definition abgebildet.

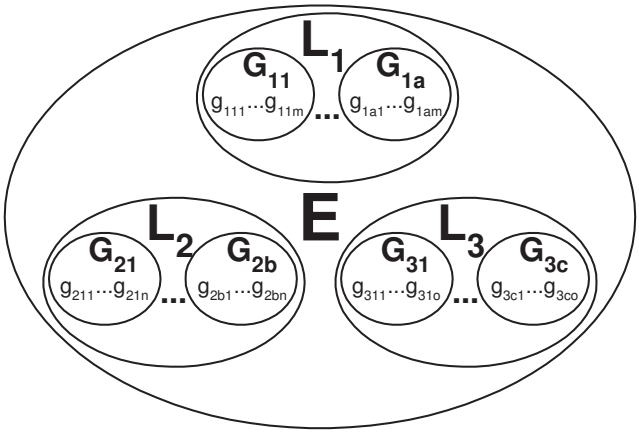

Abbildung 3.4: Euler-Vennsche Diagramm der Teilengen G_1, G_2 und G_3

3.2.1 Klassifizierung von Modelleigenschaften

In diesem Abschnitt sollen die Modelleigenschaften den Eigenschaftsmengen L_1, L_2 und L_3 zugewiesen werden. Hierzu müssen Regeln und Definitionen aufgestellt werden, die es erleichtern eine Klassifizierung vorzunehmen.

Definition 3.3 (Eigenschaften für Menge L_1)
Hier sollen alle Eigenschaften zusammengefasst werden, die bei einer Modellbeschreibung auf hoher Abstraktionsebene verwendet werden. Hierzu zählen Beispielsweise die Verstärkung eines OpAmps oder die Verzögerungszeit zwischen Ein- und Ausgang bei einer Treiberstufe. Diese Eigenschaften beschreiben das Verhalten des ICs oder des Blocks ohne Kenntnis der internen Realisierung.

Definition 3.4 (Eigenschaften für Menge L_2)
In L_2 werden alle Eigenschaften, die bei einer Modellbeschreibung auf einer niedrigeren Abstraktionsebene als bei der Menge L_1 definiert sind, zusammengefasst. Hierzu zählen beispielsweise der Ausgangswiderstand eines OpAmps oder die Slew-Rate bei einer Treiberstufe.

Definition 3.5 (Eigenschaften für Menge L_3)
In dieser Menge werden alle Eigenschaften der niedrigsten Abstraktionsebene zusammengefasst. Hierzu zählen beispielsweise Eigenschaften eines Transistormodells, welches im Zuge einer Mixed-Level Modellierungsmethodik (siehe 2.3.1) benötigt wird.

In Bild 3.5 sind die drei Mengen L_1, L_2 und L_3 in Abhängigkeit der Abstraktionsebene und somit des Detaillierungsgrades dargestellt. Die Modelleigenschaften der Menge L_1 haben den höchsten und die Eigenschaften der Menge L_3 den niedrigsten Abstraktionsgrad.

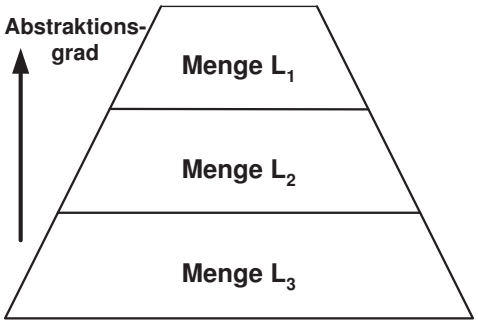

Abbildung 3.5: Abstraktionsgrad der Menge L_1, L_2 und L_3

Die Literatur schlägt in dem Bereich der Abstraktionsebenen (siehe Kapitel 2.1) mehr als drei Ebenen vor. In [Hus01A] werden z.B. fünf Ebenen vorgestellt, die von der Funktionalen- bis hin zur Transistorebene bzw. Layoutebene reicht. Diese Arbeit beschränkt sich auf die Einführung von nur drei Ebenen. Diese drei Ebenen entsprechen dem Vorschlag aus [Hus01A] für

Mixed-Signal Abstraktionsebenen den Ebenen der Verhaltensmodellen (Algorithmisch, Prozedural, Komponente). Für rein analoge Betrachtungen steht die Menge L_1 für die Eigenschaften aus der Funktionalen- und der Verhaltensebene, die Menge L_2 für die Verhaltens- und die Makromodellebene und die Menge L_3 für die Makromodell- und die Transistorebene. Nach der

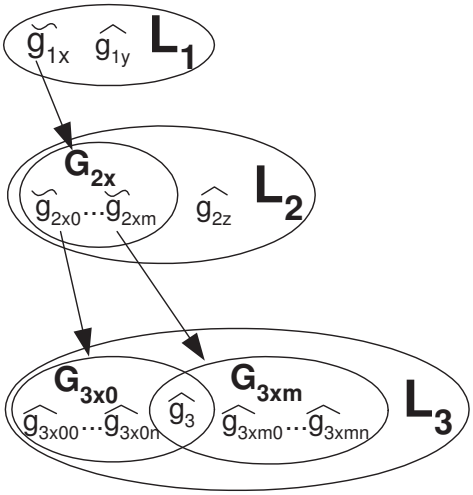

Abbildung 3.6: Abhängigkeit der Modelleigenschaften

Klassifizierung der Eigenschaften wird sichtbar, dass sich „ähnliche“ Modelleigenschaften in allen drei Mengen wiederfinden (siehe Bild 3.6). Für die Teilmengen soll die selbe Definition wie für die Mengen L_1 bis L_3 gelten, die wie folgt aussieht:

Definition 3.6
$\widehat{g_x}$ = ähnliche Modelleigenschaft auf einer tieferen Level-Ebene vorhanden
$\widetilde{g_x}$ = keine ähnliche Modelleigenschaft auf einer tieferen Level-Ebene vorhanden

Weiter soll gelten:

$$\widehat{g_1} \in \widehat{L_1} \tag{3.37}$$
$$\widetilde{g_1} \in \widetilde{L_1} \tag{3.38}$$
$$\widehat{g_2} \in \widehat{L_2} \tag{3.39}$$
$$\widetilde{g_2} \in \widetilde{L_2} \tag{3.40}$$
$$\widehat{g_3} \in \widehat{L_3} \tag{3.41}$$
$$\widetilde{g_3} \in \widetilde{L_3} \tag{3.42}$$
$$\tag{3.43}$$

Da L_3 die niedrigste Abstraktionsebene darstellt, ergibt sich:

$$\widetilde{L_3} = \{\} \tag{3.44}$$

und somit ist auch

$$\widetilde{G_3} = \{\} \tag{3.45}$$

Die Modelleigenschaft $g_{1x} \in L_1$ wird in der Menge L_2 durch die Teilmenge G_{2x} mit $G_2 \subset L_2$ mit $(g_{2x1}...g_{2xm}) \in G_2$ beschrieben. In Menge L_3 wird die Eigenschaft g_{2x1} wiederum durch die Teilmenge G_{3x1} mit $G \subset L_3$ mit $(g_{3x11}...g_{3x1n}) \in G_3$ zusammengefasst.

Beispiel:
Das nachfolgende Bild (siehe Bild 3.7) soll die Abhängigkeiten der Modelleigenschaften in den drei Eigenschaftsmengen verdeutlichen. Die Verzögerungszeit einer Ausgangsstufe t_d stellt hier beispielhaft eine Modelleigenschaft von Menge L_1 dar. Diese Modelleigenschaft wird in Menge L_2 durch die Teilmenge G_2 beschrieben. Die Teilmenge beinhaltet z.B. den Ausgangswiderstand

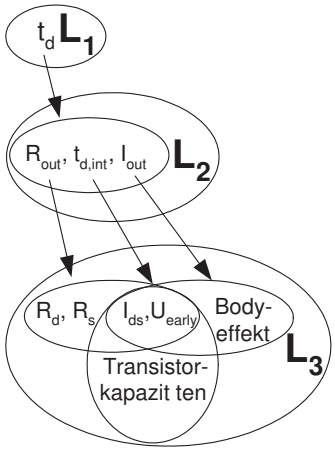

Abbildung 3.7: Beispiel der Abhängigkeiten von Modelleigenschaften

R_{out}, den Ausgangsstrom I_{out} und eine interne Verzögerungszeit $t_{d,int}$. In der Menge L_2 werden je nach Realisierung des Modells, die in Menge L_1 beschriebene Eigenschaften, durch mehrere Eigenschaften repräsentiert. Ist das Modell im Zuge einer Mixed-Level Modellierung, wie in Kapitel 2.3 vorgestellt entstanden, sind die Modelleigenschaften abhängig von den Transistormodelleigenschaften wie Gate- und Substratkapazitäten, Earlyeffekt, Bahnwiderstand, Substratdioden und dem Drainstrom I_d (abhängig von k_p). So wird beispielsweise $t_{d,int}$ durch I_d , Earlyeffekt und den Transistorkapazitäten realisiert.

3.3 Herleitung der Modellierungsmethodik

In diesem Abschnitt werden Modellierungsmethoden vorgestellt, die es ermöglichen die zuvor beschriebenen Modelleigenschaften bei der Implementierung zu berücksichtigen. Bei der Suche nach Lösungen sollte hierbei der gesamte Entwicklungs-Flow mit seinen verschiedenen Bereichen (z.B. Testentwicklung oder Schaltungsentwicklung) im Fokus stehen. Auch innerhalb der einzelnen Bereiche müssen verschiedene Strategien berücksichtigt werden (z.B. die Designmethodik, siehe Kapitel 2.2).

Eine weitere Anforderung an die Methodik stellt die zeitliche Verfügbarkeit von Modelleigenschaften im Entwicklungsprozess dar (siehe Bild 3.8). Zu Beginn der IC-Entwicklung stehen Systembetrachtungen im Vordergrund. In dieser frühen Phase gibt es nur wenige Modelleigenschaften, die zu berücksichtigen sind. Die Anzahl der Eigenschaften werden im Laufe der Entwicklung, wie in Bild 3.8 skizziert, zunehmen. Wird die Bottom-Up-Modellierungsmethodik angewandt, sollten alle Modelleigenschaften, die zur IC-Entwicklung beitragen, zur Verfügung stehen.

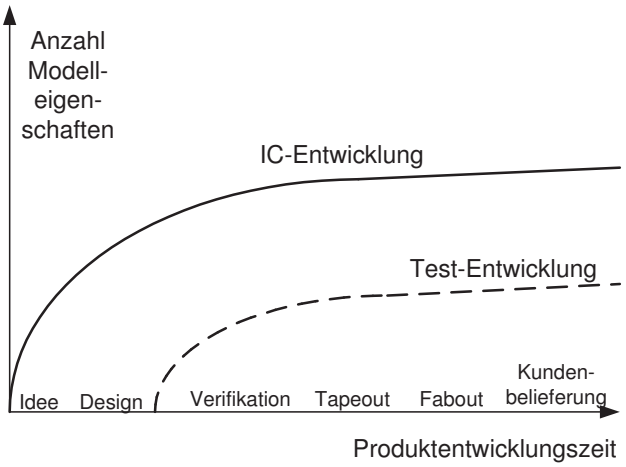

Abbildung 3.8: Anzahl der Modelleigenschaften im Laufe des Entwicklungsprozesses

Deshalb sollten Modelle modular aufgebaut sein, damit Modelleigenschaften, die zu Beginn der Modellierung nicht zu Verfügung stehen, nachträglich leicht bereit gestellt werden können. Die zu entwickelnde Modellierungsmethodik sollte dieses berücksichtigen.

Bei der Analyse verschiedener Schaltungen derselben Schaltungsklasse (z.B. Automotive-ICs) wird deutlich, dass immer wieder bestimmte Schaltungskomponenten zum Einsatz kommen. Diese können zum Teil direkt oder mit leichten Veränderungen in den verschiedenen Schaltungen auftreten. Diese Schaltungskomponenten sollten als Verhaltensmodelle in einer Bibliothek

angelegt werden, mit dem Ziel, bei Nachfolgeprojekten die Modellierungszeit erheblich zu verkürzen.

Bei der Realisierung der Modellierungsmethodik sollen alle die zuvor beschriebenen Randbedingungen mit berücksichtigt werden. Der Schwerpunkt dieser Arbeit richtet sich somit auf konfigurierbare Verhaltensmodelle, die den verschiedenen Anforderungen im Entwicklungsprozess und die zeitliche Verfügbarkeit von Modelleigenschaften Rechnung tragen. Im nachfolgenden Abschnitt werden die erarbeiteten Lösungen näher beschrieben.

3.3.1 Einführung verschiedener Modell-Level

Wünschenswert ist es ein Verhaltensmodell zu entwickeln, das alle Modelleigenschaften $m_{ME,n} \subseteq L_1 \cup L_2 \cup L_3$ beinhaltet und diese nach Bedarf ein- und ausschaltbar sind. Dies lässt sich in den meisten Fällen nicht realisieren, weil die Basis für einige Modelleigenschaften bestimmte Modelltopologien voraussetzen. Soll das Modell z.B. in einem Top-Down-Design eingesetzt werden, fehlen bei der Systementwicklung wesentliche Topologieinformationen und Modelleigenschaften der zu modellierenden Komponente.

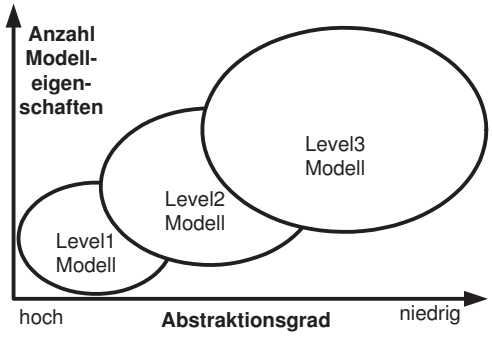

Abbildung 3.9: Abstraktionsgrad der verschiedenen Modell-Level

Die Lösung hierfür ist die Einführung verschiedener Abstraktionsebenen, die sich in ihrer Topologie sowie im Detaillierungsgrad unterscheiden. Diese Klassen werden als „Level1-Modell“ für die höchste bis „Level3-Modell“ für die niedrigste betrachtete Abstraktion bezeichnet (siehe Bild 3.9).

Diese Art der Modellierungsmethodik ist optimal für eine Top-Down-Design Entwicklung geeignet. Bild 3.10 soll die zeitliche Realisierung der Level1 bis Level3-Modelle bei der IC-Entwicklung, mit zunehmender Anzahl der Modelleigenschaften, darstellen.

Die nachfolgende Tabelle 3.1 soll beispielhaft verdeutlichen, mit welchen Komponenten ein Schaltverhalten (z.B. bei einer Treiberstufe) modelliert werden kann.

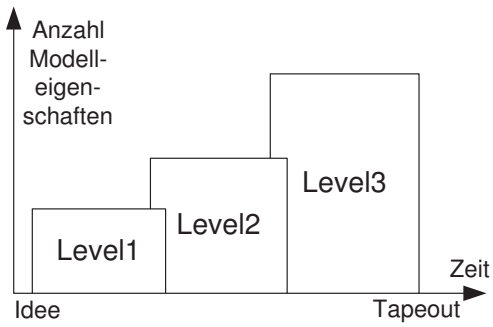

Abbildung 3.10: Modellerstellung der Level1 bis Level3-Modelle bei der IC-Entwicklung

Detaillierung	**Komponente**	**Nachbildung des Schaltverhaltens**
Level3	aktiv	Parametrisierte Komponenten
Level2	aktiv	Parametrisierte Komponenten (reduziert)
Level1	passiv	RC-Glied

Tabelle 3.1: Modellnachbildung einer Treiberstufe

Definition 3.7 (Eigenschaften für Level1-Modelle)
Die Level1-Modelle sollen alle Modelleigenschaften $m_{ML1,a}$ der Menge L_1 beinhalten. Diese Modelle können zu einer frühen Phase bei der IC-Entwicklung erstellt werden (z.B. Top-Down-Designmethodik)

$$M_{ML1} = \left\{ m_{ML1} | m_{ML1,1}, ..., m_{ML1,a} \right\} \\ wobei: \; m_{ML1,1}, ..., m_{ML1,a} \in L_1 \tag{3.46}$$

Definition 3.8 (Eigenschaften für Level2-Modelle)
Das Level2-Modell beinhaltet alle Modelleigenschaften aus Menge L_2. Weiter sollen zusätzlich noch Modelleigenschaften aus Menge $\widehat{L_1}$, die nicht im Widerspruch zur Topologie und zu den Eigenschaften aus L_2 stehen, implementiert werden. Da die Modelleigenschaften aus Menge $\widetilde{L_1}$, wie in Abschnitt 3.2.1 beschrieben, „ähnlich“ zu Menge L_2 sind, dürfen diese aus L_1 nicht

berücksichtigt werden.

$$M_{ML2} = \left\{ m_{ML2} | m_{ML2,1}, ..., m_{ML2,b} \right\}$$
$$wobei: \ (m_{ML2,1}, ..., m_{ML2,b} \in L_2 \cup \widehat{L_1}) \ \cap$$
$$(m_{ML2,1}, ..., m_{ML2,b} \notin \widetilde{L_1}) \quad (3.47)$$

Definition 3.9 (Eigenschaften für Level3-Modelle)
Die Modelleigenschaften aus Menge L_3 beinhaltet das Level3-Modell. Weiter sollen wie im Level2-Modell zusätzlich Modelleigenschaften aus Menge $\widehat{L_1}$ und $\widehat{L_2}$, die nicht im Widerspruch zur Topologie und zu den Eigenschaften aus der Menge L_3 stehen, implementiert werden. Diese Modelle können erst in einer späteren Phase der IC-Entwicklung erstellt werden (z.B. bei einer Bottom-Up-Methodik)

$$M_{ML3} = \left\{ m_{ML3} | m_{ML3,1}, ..., m_{ML3,c} \right\}$$
$$wobei: \ (m_{ML3,1}, ..., m_{ML3,c} \in L_3 \cup \widehat{L_2} \cup \widehat{L_1}) \ \cap$$
$$(m_{ML3,1}, ..., m_{ML3,c} \notin \widetilde{L_2} \cup \widetilde{L_1}) \quad (3.48)$$

In dem nachfolgenden Abschnitt soll die Modellstruktur und deren Modellierungsmethodik für die drei verschiedenen Abstraktionsebenen vorgestellt werden. Die Eigenschaften sollen hierbei wie zuvor beschrieben berücksichtigt werden.

Level3-Modell

Bei einem Level3-Modell wird das Modell in einer Mixed-Level und Mixed-Signal Modellierung realisiert (siehe Kapitel 2.3.1). Diese Art der Modellierung stellt die genaueste Modellierungsmethode und somit die niedrigste Abstraktionsebene dar. Ziel dieser Methode ist es, ein möglichst identisches Verhalten der originalen Transistorschaltung an den Ausgängen bzw. Eingängen nachzubilden. Dafür sorgen in erster Linie die Verwendung von Transistoren, die sich direkt an den Ein- bzw. Ausgängen befinden und somit eine realistische Funktion an der Peripherie realisieren. Ebenso wird beispielsweise die Ansteuerung dieser Ausgangstransistoren auf Transistorebene implementiert, so dass ein möglichst originales Verhalten erzielt wird. Diese Transistoren enthalten auf Transistorebene meist BSIM3, EKV oder einfache Spice Modelle, die durch W und L skalierbar sind. Die zuvor beschriebenen Modelleigenschaften werden in dem Modell implementiert.
Zusammenfassend lassen sich folgende Regeln ableiten:

Regeln für Level3:

- Verwendung der originalen Transistoren an der Peripherie des ICs
- Originale Ansteuerung der Peripherie- Transistoren beibehalten
  - zur Gatestrom-Regulierung (z.B. für Slew-Rate-Kontrolle)
  - interne Regelkreise steuern den Gatestrom (z.B. Linearregler)
  - Stromspiegel an Drain oder Source (z.B. für Strombegrenzung)
- Interne Funktionsblöcke werden idealisiert mit einer HDL nachgebildet
  - Stromspiegel
  - OpAmps
  - Differenzstufen
  - Spannungspegel-Wandlung (=Levelshifter)
  - Linearregler
  - Verzögerungsstufen (Delay-Stufen)

Level2-Modell

Beim Level2-Modell handelt es sich um eine Vereinfachung des Level3-Modells in dem versucht wird, interne Transistoren durch idealisierte Verhaltensmodelle zu ersetzen. Die Verwendung der Transistoren an der Peripherie bleiben wie im Level3-Modell weiterhin bestehen, nur dessen Ansteuerung werden durch HDL-Modelle idealisiert. So werden beispielsweise Buffer-Stufen, die zum Regulieren der Gateströme eingesetzt werden, durch ideale Strombegrenzer, die in einer HDL realisiert sind, ersetzt. Eine weitere Vereinfachung ist, dass interne analoge in digitale Signale überführt werden können. Hierdurch stellt sich ein messbarer Performancegewinn, sowie durch den Wegfall weiterer Transistoren im Vergleich zum Level3-Modell, ein. Diese Modellstruktur wird als Level2-Modell bezeichnet. Das Verhalten bei der Simulation ist nahezu identisch zur Level3 Simulation. Geringe Abweichungen ergeben sich beispielsweise beim Verhalten einer Slew-Rate oder beim Temperaturverhalten. Grund hierfür ist die vereinfachte Beschreibung interner Funktionen.

Zusammenfassend lassen sich folgende Regeln ableiten:

Regeln für Level2:

- Verwendung der originalen Transistoren an der Peripherie des ICs
- idealisierte Ansteuerung der Peripherie-Transistoren durch HDL-Modelle
- interne Funktionsblöcke werden idealisiert mit einer HDL nachgebildet
- interne analoge Signale, wenn möglich, in digitale Signale überführen

Level1-Modell

Der Einsatz des Level1-Modells findet auf höchster Abstraktionsebene statt. Es werden nur passive Elemente bei der Modellierung verwendet, die alle in einer Mixed-Signal Hardwarebeschreibungssprache notiert sind. Befindet sich die zu modellierende Komponente an der Peripherie des ICs (z.B. Ausgangstreiberstufe), so wird der Ausgangstransistor in seiner Grundfunktion durch einen digital umschaltbaren Widerstand (Schalter) realisiert, der je nach Zustand des Digitalsignals zwischen $R_{DS,off}$ und $R_{DS,on}$ wechselt. Für die Kanalkapazitäten des Transistors werden externe Kapazität hinzugefügt, die so dimensioniert werden müssen, dass sich ein idealisiertes Verhalten mit der originalen Transistorschaltung ergibt. Dies ist nur bedingt realisierbar, da das RC Verhalten meist nicht mit einem Slew-Rate Verhalten eines MOS-Transistors übereinstimmt. Für die Messtechnik ist z.B. in den meisten Fällen nur die Zeit zwischen 10% und 90% des Endsignalzustands von Bedeutung. Diesbezüglich lässt sich das RC-Verhalten so dimensionieren, dass die in der Spezifikation festgelegte zu messende Zeit, erreicht wird.

Ein weiteres externes Element des Level1-Modells ist die Diode, welche die Bulk-Drain-Diode des MOS-Transistors nachbildet. Diese Modellanforderung kommt vom virtuellen Test her und wird beispielsweise bei OPEN/Short-Messungen benötigt. Als Messergebnis wird eine positive und eine negative Diodenflussspannung erwartet, die durch die Diode im Level1-Modell und durch die Bulk-Drain-Diode im Level2 und Level3-Modell nachgebildet wird. Interne Komponenten die beispielsweise für die Ansteuerung der Schalter verantwortlich sind, werden vereinfacht, wenn möglich mit digitalen Signalen, nachgebildet.
Zusammenfassend lassen sich folgende Regeln ableiten:

Regeln für Level1:

- nur passive Elemente und idealisierte HDL-Modell werden verwendet

- Transistoren werden durch digital gesteuerte Schalter ($R_{DS,on}$ und $R_{DS,off}$) ersetzt
- Einfügen von Dioden (=Bulk-Drain-Dioden) an der Peripherie
- Einfügen von Kapazitäten (=Kanalkapazitäten) an der Peripherie
- interne Komponenten oder Signale, wenn möglich, digital realisieren

3.3.2 Bibliothekskonzept für Verhaltensmodelle

Wie bereits erwähnt, ist es wünschenswert das Verhaltensmodell in kürzester Zeit, am besten durch eine Art Modulbaukasten, zu erstellen. In Rahmen dieser Arbeit ist eine Bibliothek erstellt worden, die häufig auftretende Schaltungsblöcke beinhaltet und somit den Baukasten zur Generierung eines Verhaltensmodells darstellt. Bei der Neuerstellung eines Bibliothekselementes, soll die zuvor beschriebene Modellierungsmethodik angewandt werden. Hierzu müssen alle Modelleigenschaften parametrisch zur Verfügung stehen.

Im Laufe dieser Arbeit sind durch Analyse mehrerer Automotive-Schaltkreise eine Vielzahl von Bibliothekselementen identifiziert und realisiert worden. Ein großer Anteil an ReUse findet sich bei den Pad- bzw. Schutzstrukturen wieder. Diese befinden sich an allen Ein- und Ausgangspins in verschiedensten Variationen, von Längs- bis Querregler, von Hochvoltschutz- bis hin zu Niedervoltschutzstrukturen.
Eine weiter häufig verwendete Komponente ist die Treiberstufe, die meist als High-Side oder Low-Side Treiber eingesetzt wird. Eingangsstufen sind ebenfalls eine immer wieder vorkommende Komponente, die sich meist nur in den verschiedenen Pegeln, z.B. CMOS oder TTL sowie deren Empfindlichkeit unterscheiden.
Diese Komponenten sind in den meisten Fällen direkt an den Pins zu finden und somit für den VT von enormer Bedeutung. Weitere Standardkomponenten, die sich im Inneren des Schaltkreises befinden, wie beispielsweise Bandgaps, OpAmps, Komparatoren, Supply-Units, SPI usw., sollten für eine effektive Modellierung natürlich ebenso als Bibliothekselement zur Verfügung stehen.

Variantenbildung

Bei der Vielzahl von Anwendungsfällen eines Bibliotheksmodells sind verschiedene Modifikationen unumgänglich. Aus diesem Grund werden die zuvor beschriebenen Modell-Level (Level1 bis Level3) in verschiedenen Varianten bereitgestellt. Die Definition einer Variante ist nicht die Änderung eines Parameters, sondern eine Topologieänderung im Modell. Die Ursachen hierfür

sind sehr vielfältig und müssen von Block zu Block neu ermittelt werden. Meist werden die zusätzlichen Varianten durch Anforderungen verursacht, die nur durch leichte Modifikation des Bibliothekselementes, erzielt werden können. Hierzu können beispielsweise Pull-Up oder Pull-Down Widerstände an den Ein- oder Ausgängen eines Bibliotheksmodells die Ursache sein. Die Notwendigkeit die Topologie zu verändern, ist beim Level3-Modell häufiger als beim Level2 und Level1-Modell zu sehen.

3.3.3 Konfigurierbare HDL-Modelle

Die Einführung von konfigurierbaren HDL-Modellen stellt eine weitere Methode dar, mit dem Ziel, Modelleigenschaften zu aktivieren oder zu deaktivieren, um einen Performancegewinn zu erhalten. Der Vorteil gegenüber den zuvor beschriebenen Modell-Level und Varianten ist die Flexibilität, mit der Modelleigenschaften je nach Bedarf ein- oder ausgeschaltet werden können. Ebenso kann mit konfigurierbaren Modellen der Abstraktionsgrad von der Transistor- bis hin zur Verhaltensebene, durch die Konfiguration des Modells erreicht werden. Mit dieser Methode können bereichsübergreifende Modelle, die unter anderem für die Verifikation der Schaltung (Bottom-Up-Methodik) als auch für virtuelle Test Anwendungen, eingesetzt werden. Für einen Einsatz im Top-Down-Design ist die Methodik weniger geeignet, da hier alle Modelleigenschaften und somit auch die Topologie in einer frühen Phase der Modellerstellung bekannt sein sollten. Das konfigurierbare Modell soll alle Modelleigenschaften aus

$$E = \{e|e_1, ..., e_n\}$$

beinhalten.
Diese Modelleigenschaften müssen durch geeignete Sprachkonstrukte gezielt deaktiviert werden können.

```
_____________________________________________
wenn Modelleigenschaft = aktiv              |
      - Modelleigenschaft aktivieren        |
sonst                                       |
      - Modelleigenschaft deaktivieren      |
ende                                        |
____________________________________________|
```

Wenn Modelleigenschaften sowohl in Menge L_1 (=$\widetilde{L_1}$), L_2 (=$\widetilde{L_2}$) und bzw. oder L_3 enthaltet sind, also in verschiedener Detaillierungstiefe (siehe Kapitel 3.2), müssen geschachtelte Sprachkonstrukte zum Aktivieren oder zum Deaktivieren verwendet werden. Durch die Konfiguration innerhalb des Modells können somit Mixed-Level Verhaltensmodelle einfach realisiert werden.

Je nach Simulationsaufgabe besteht hier die Möglichkeit gezielt Eigenschaften in verschiedenen Detaillierungstiefen auszuwählen.

```
_____________________________________________
wenn Modelleigenschaft = aktiv               |
                                             |
      wenn Detaillierungstiefe = Level1      |
                                             |
           - ME aus Menge L1 aktivieren      |
                                             |
      wenn Detaillierungstiefe = Level2      |
                                             |
           - ME aus Menge L2 aktivieren      |
                                             |
      sonst Level3                           |
                                             |
           - ME aus Menge L3 aktivieren      |
                                             |
      ende wenn/sonst                        |
sonst                                        |
                                             |
      - keine Funktion                       |
ende wenn/sonst                              |
_____________________________________________|
```

Wie bereits in Kapitel 3.2 dargestellt, werden Modelleigenschaften in Menge $\widetilde{L_1}$ meist durch mehrere Eigenschaften, z.B. aus der Menge L_2, die dann als Teilmenge $G_2 \subset L_2$ definiert ist, beschrieben.

3.3.4 Konfigurationsmöglichkeiten bei Level1 bis Level3-Modellen

In diesem Abschnitt soll eine Kombination aus den zuvor beschriebenen Modellierungsmethoden der Modell-Level und der konfigurierbaren Modellen vorgestellt werden.

Nachteil beim Erstellen der konfigurierbaren Modelle ist, wie in Kapitel 3.3.3 beschrieben, die Verfügbarkeit aller Modelleigenschaften und das Wissen über Schaltungstopologie der zu modellierenden Komponenten. Bei einem Top-Down-Design sind diese Voraussetzungen in den meisten Fällen nicht gegeben, im Gegensatz hierzu werden diese bei einer Bottom-Up-Strategie

jedoch erfüllt. Der Vorteil der konfigurierbaren Modelle ist das Ein- und Ausschalten von Modelleigenschaften, die z.B. von der Simulationsaufgabe gefordert werden, mit dem Ziel eine Performancesteigerung zu erhalten.

Die Level1 bis Level3-Modelle bieten diesen Vorteil zum Deaktivieren von Modelleigenschaften nicht. Dafür kann diese Modellierungsmethodik für ein Top-Down-Design verwendet werden. In der nachfolgenden Tabelle 3.2 sind nochmals die Vor- und Nachteile gegenübergestellt.

	Level-Modelle	**konfigurierbare Modelle**
Top-Down	optimal	eingeschränkt möglich
Bottom-Up	ja	optimal
Meet-In-The-Middle	ja	eingeschränkt möglich
Änderung der Topologie	nein	ja
Ein/Ausschalten von ME	nein	ja
geeignet für VT	ja	ja

Tabelle 3.2: Eigenschaften der Modellierungsmethoden

Das getrennte Anwenden beider Modellierungsmethoden bei einer Produktentwicklung würde einen nicht zu vernachlässigenden Mehraufwand bedeuten. Sollen mit unter Verhaltensmodelle nicht beim Top-Down-Design sondern nur bei der Verifikation der Schaltung und bei dem VT eingesetzt werden, würden sich die konfigurierbaren Modelle anbieten.

Ist es nun das Ziel, dass die Verhaltensmodelle zusätzlich die IC-Entwicklung mit den Entwurfsmethoden wie Top-Down oder Meet-In-The-Middle unterstützen, muss eine „effizientere“ Methode angewandt werden. Hierzu gibt es zwei Lösungen:

- Konfigurierbare Level1-Level3-Modelle
- Austauschen einzelner Subblöcke der Level1-Level3-Modelle durch konfigurierbare Modelle

Bei den konfigurierbaren Level1-Level3-Modellen sollen folgende Definitionen bei der Modellerstellung gelten.

Definition 3.10 (Eigenschaften der konfigurierbaren Level1-Modelle)
Die Level1-Modelle sollen alle Modelleigenschaften der Menge L_1 beinhalten. (siehe Kapitel 3.2)

$$M_{ML1,konfig} = \left\{ m_{ML1,konfig} | m_{ML1,konfig,1}, ..., m_{ML1,konfig,d} \right\} \tag{3.49}$$

$$m_{ML1,konfig,1}, ..., m_{ML1,konfig,d} \in L_1 \tag{3.50}$$

Definition 3.11 (Eigenschaften der konfigurierbaren Level2-Modelle)
Das Level2-Modell beinhalten alle Modelleigenschaften aus Menge L_2. Weiter sollen zusätzlich noch Modelleigenschaften aus Menge L_1, die nicht im Widerspruch zur Topologie stehen, implementiert werden. Da die Modelleigenschaften aus Menge L_1, wie in Abschnitt 3.2.1 beschrieben, „ähnlich“ zu Menge L_2 sein können (=$\widetilde{L_1}$), müssen diese bei der Modellrealisierung austauschbar, d.h. konfigurierbar sein.

$$M_{ML2,konfig} = \left\{ m_{ML2,konfig} | m_{ML2,konfig,1}, ..., m_{ML2,konfig,e} \right\} \tag{3.51}$$

$$m_{ML2,konfig,1}, ..., m_{ML2,konfig,e} \in L_2 \cup \widehat{L_1} \cup \widetilde{L_1} \tag{3.52}$$

Definition 3.12 (Eigenschaften der konfigurierbaren Level3-Modelle)
Die Modelleigenschaften aus Menge L_3 beinhalten das Level3-Modell. Weiter sollen wie im Level2-Modell zusätzlich Modelleigenschaften aus Menge L_1 und L_2, die nicht im Widerspruch zur Topologie stehen, implementiert werden. Bei der Modellerstellung müssen „ähnliche“ Eigenschaften (=$\widetilde{L_1}$ und $\widetilde{L_2}$) umschaltbar sein.

$$M_{ML3,konfig} = \left\{ m_{ML3,konfig} | m_{ML3,konfig,1}, ..., m_{ML3,konfig,f} \right\} \tag{3.53}$$

$$m_{ML3,konfig,1}, ..., m_{ML3,konfig,f} \in L_3 \cup \widehat{L_2} \cup \widetilde{L_2} \cup \widehat{L_1} \cup \widetilde{L_1} \tag{3.54}$$

Mit dieser Methode werden zusätzlich die Eigenschaften der Modelle der höheren Abstraktionsebenen mit implementiert. Das Level3-Modell enthält somit alle Modelleigenschaften der Menge E. In der Praxis lässt sich diese Methode jedoch mehr auf Subblöcke wie beispielsweise OpAmps, Komparatoren, Filter oder Ausgangsstufen optimal anwenden. Die Begründung hierfür liegt in der Topologie, die sich im Laufe des Entwicklungsprozess verändern kann. Bei Subblöcken ist diese noch überschaubar und lässt sich im Source-Code wie in Kapitel 3.3.3 beschrieben realisieren. Für größere Schaltungen, wie es etwa ein DC-DC Aufwärtswandler darstellt, sind diese Topologieänderungen von Level1 bis hin zum Level3-Modell nicht mehr mit dieser Methode zu realisieren. Für diese Art von Schaltungen ist dann ein Austauschen der Subblöcke des

Level3-Modells durch konfigurierbare Modelle sinnvoll. Im nachfolgenden Abschnitt wird dies näher beschrieben.

3.3.5 Konfigurierbare Level1 bis Level3-Modelle im Entwurfsablauf

Bild 3.11 soll qualitativ den Entwicklungsablauf eines Blockes größerer Komplexität (z.B. DC-DC Aufwärtswandler) und somit den Ablauf bei der Modellerstellung verdeutlichen.

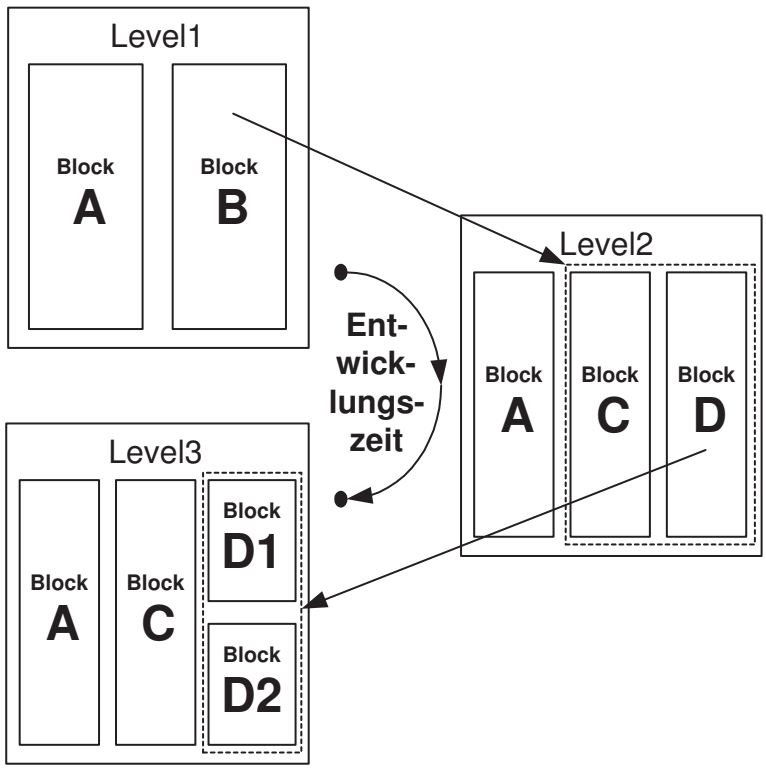

Abbildung 3.11: Topologieveränderungen im Entwurfsprozess

Zu Beginn der Systemen- bzw. Architekturentwicklung wird das Level1-Modell mit den Eigenschaften aus L_1, welches die Subblöcke A und B beinhaltet, entwickelt. Im nächsten Entwurfsschritt wird das Level2-Modell erstellt, bei dem sich topologisch der Subblock B in zwei Subblöcke C und D unterteilt. Der Subblock A ist von seiner Topologie gleich geblieben, sodass die Eigenschaften aus dem Level1-Modell hier mit einfließen können. Somit enthält Subblock A die Eigenschaften der Mengen L_1 und L_2. Für die Subblöcke B und C müssen nun die Eigenschaften nach den Definitionen der Mengen L_1, L_2 und L_3 neu ermittelt werden. Danach kann für

jeden Block ein konfigurierbares Modell mit den Modelleigenschaften aus L_1 und L_2 entwickelt werden. Im Bild 3.11 ist der Übergang vom Subblock B des Level1-Modells zu den Subblöcken C und D des Level2-Modells verdeutlicht.

Die Menge $L_{1,B}$ wird in zwei Teilmengen $L_{1,C}$ und $L_{1,D}$ unterteilt, für diese gilt:

$$L_{1,B} = L_{1,C} \cup L_{1,D} \tag{3.55}$$

Die Eigenschaften der Menge L_2 müssen für Subblock C und D neu ermittelt werden.

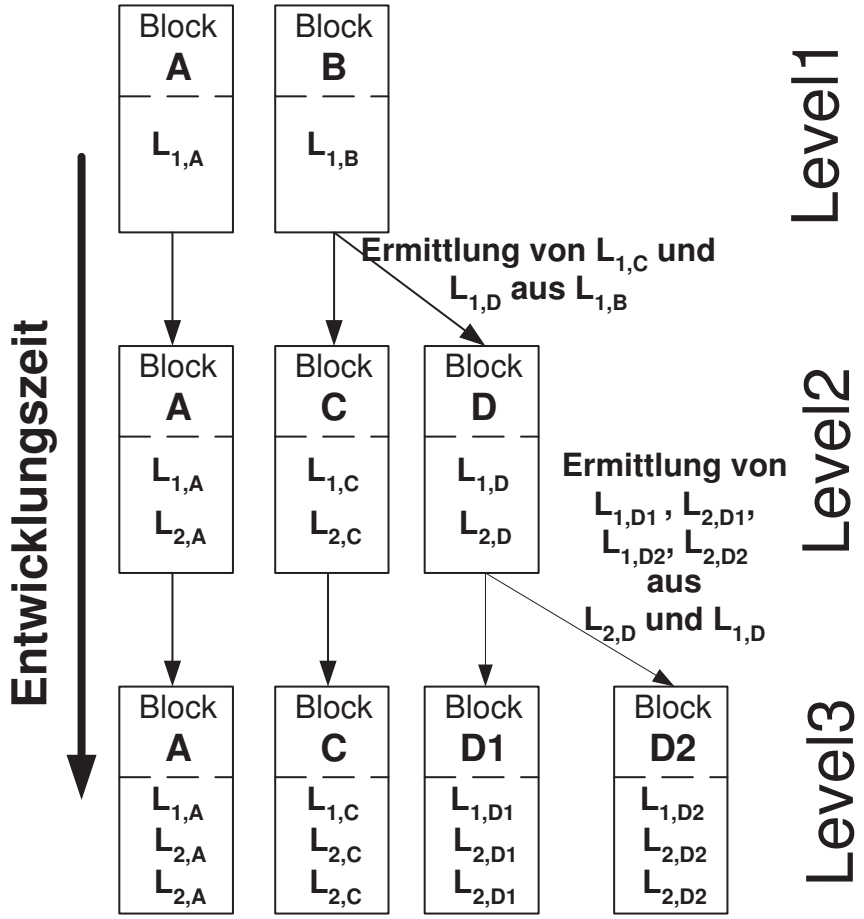

Abbildung 3.12: Modelleigenschaften im Entwurfsprozess

Der nächste Übergang vom Level2 zum Level3-Modell beinhaltet wieder eine Topologieänderung. Subblock D vom Level2-Modell wird im Level3-Modell zu D_1 und D_2. Block A bleibt topologisch unverändert und kann somit die Eigenschaften von L_3, L_2 und L_1 beinhalten. Ebenso wie Subblock C der topologisch zum Level2-Modell unverändert bleibt und dessen Eigenschaften sich aus L_{1C}, L_2 und L_3 zusammensetzen. Wegen der Topologieänderung vom Subblock D

müssen die Eigenschaften $L_{1,D}$ und L_2 unterteilt werden, für die folgendes gilt:

$$L_{1,D} = L_{1,D1} \cup L_{1,D2} \tag{3.56}$$

$$L_{2,D} = L_{2,D1} \cup L_{2,D2} \tag{3.57}$$

In Bild 3.12 sind die Modelleigenschaften für die jeweiligen Topologieänderungen im Entwurfsprozess dargestellt. Für die einzelnen Modelle (A,B,C,D,D1 und D2) gelten folgende Modellmengen der unterschiedlichen Modell-Level:

Level1:

$$M_{A,ML1,konfig} = \left\{ m_{A,ML1,konfig} | m_{A,ML1,konfig,1}, ..., m_{A,ML1,konfig,g} \right\}$$
$$m_{A,ML1,konfig,1}, ..., m_{A,ML1,konfig,g} \in L_1 \tag{3.58}$$

$$M_{B,ML1,konfig} = \left\{ m_{B,ML1,konfig} | m_{B,ML1,konfig,1}, ..., m_{B,ML1,konfig,h} \right\}$$
$$m_{B,ML1,konfig,1}, ..., m_{B,ML1,konfig,h} \in L_1 \tag{3.59}$$

Level2:

$$M_{A,ML2,konfig} = \left\{ m_{A,ML2,konfig} | m_{A,ML2,konfig,1}, ..., m_{A,ML2,konfig,i} \right\}$$
$$m_{A,ML2,konfig,1}, ..., m_{A,ML2,konfig,i} \in L_{2,A} \cup L_{1,A} \tag{3.60}$$

$$M_{C,ML2,konfig} = \left\{ m_{C,ML2,konfig} | m_{C,ML2,konfig,1}, ..., m_{C,ML2,konfig,j} \right\}$$
$$m_{C,ML2,konfig,1}, ..., m_{C,ML2,konfig,j} \in L_{2,C} \cup L_{1,C} \tag{3.61}$$

$$M_{D,ML2,konfig} = \left\{ m_{D,ML2,konfig} | m_{D,ML2,konfig,1}, ..., m_{D,ML2,konfig,k} \right\}$$
$$m_{D,ML2,konfig,1}, ..., m_{D,ML2,konfig,k} \in L_{2,D} \cup L_{1,D} \tag{3.62}$$

Level3:

$$M_{A,ML3,konfig} = \left\{ m_{A,ML3,konfig} | m_{A,ML3,konfig,1}, ..., m_{A,ML3,konfig,l} \right\}$$
$$m_{A,ML3,konfig,1}, ..., m_{A,ML3,konfig,l} \in L_{3,A} \cup L_{2,A} \cup L_{1,A} \quad (3.63)$$

$$M_{C,ML3,konfig} = \left\{ m_{C,ML3,konfig} | m_{C,ML3,konfig,1}, ..., m_{C,ML3,konfig,m} \right\}$$
$$m_{C,ML3,konfig,1}, ..., m_{C,ML3,konfig,m} \in L_{3,C} \cup L_{2,C} \cup L_{1,C} \quad (3.64)$$

$$M_{D1,ML3,konfig} = \left\{ m_{D1,ML3,konfig} | m_{D1,ML3,konfig,1}, ..., n_{D1,ML3,konfig,n} \right\}$$
$$m_{D1,ML3,konfig,1}, ..., m_{D1,ML3,konfig,n} \in L_{3,D1} \cup L_{2,D1} \cup L_{1,D1} \quad (3.65)$$

$$M_{D2,ML3,konfig} = \left\{ m_{D2,ML3,konfig} | m_{D2,ML3,konfig,1}, ..., o_{D2,ML3,konfig,o} \right\}$$
$$m_{D2,ML3,konfig,1}, ..., m_{D2,ML3,konfig,o} \in L_{3,D} \cup L_{2,D} \cup L_{1,D} \quad (3.66)$$

Das Level3-Modell beinhaltet alle Modelleigenschaften aus der Menge E und kann somit für andere Bereiche wie z.B. den VT eingesetzt werden.

In der Praxis wird immer versucht den Modellierungsaufwand zu minimieren. Wie zuvor beschrieben, bedeutet das Erstellen von konfigurierbaren Modellen für alle Subblöcke des Level2 und des Level3-Modells einen Mehraufwand gegenüber den einfachen Level1-Modellen. Jedoch hat dies den Nachteil, keinen Performacegewinn bei einzelnen Simulationsaufgaben zu bekommen. Einen Kompromiss stellt hier das gezielte „Austauschen“ der Subblöcke des einfachen Level3-Modells durch konfigurierbare Modelle dar. So können beispielsweise Blöcke, die sehr rechenintensiv und somit einen hohen Simulationsaufwand verursachen, durch konfigurierbare Modelle ausgetauscht werden. Blöcke die nur wenig Performanceverlust verursachen, gilt es somit unverändert als einfaches Level-Modell einzusetzen. Bei einer Mixed-Level Modellierung ist es möglich z.B. gezielt die Transistoren durch „konfigurierbare Transistoren“ auszutauschen. Dieses kann ohne großen Aufwand realisiert werden, wenn die konfigurierbaren MOSFET HDL-Modelle zur Verfügung stehen. Im nachfolgenden Kapitel soll die Realisierung solcher HDL-Modelle vorgestellt werden.

Kapitel 4

Realisierung der Methodik

In diesem Kapitel sollen die zuvor vorgestellten Modellierungsmethoden realisiert werden. Beginnend mit der Modellierung von Bibliothekselementen sollen die in Kapitel 3.3.1 vorgestellten Modell-Level am Beispiel einer Ausgangstreiberstufe umgesetzt werden. Weiter werden konfigurierbare HDL-Modelle und im speziellen konfigurierbare MOSFET HDL-Modelle zur Realisierung gebracht. Abgeschlossen wird dieses Kapitel mit der Kombination beider Methoden und einem begleitenden Modellierungsbeispiel.

4.1 Realisierung von Level1 bis Level3-Modellen

Nachfolgend werden die drei verschiedenen Modell-Level und deren Varianten am Beispiel einer Ausgangstreiberstufe demonstriert. Diese Modelle sollen als Bibliothekselemente aufbereitet werden, sodass sie jederzeit wieder verwendet werden können. Es handelt sich hierbei um eine Bottom-Up-Vorgehensweise, bei der die Schaltung bzw. die Topologie und alle Modelleigenschaften ME aus der Menge E bekannt sind.

In diesem Beispiel wird eine Ausgangsstufe, die für mittlere bis hohe Ausgangsleistungen (von 1mA bis 1A) ausgelegt ist, verwendet. Ein häufiges Einsatzgebiet von Ausgangstreiberstufen ist die Ansteuerung von H-Brücken oder externen Power-MOSFETs. Die DMOS-Ausgangstransistoren des ICs sind sehr gross und ihre Schaltgeschwindigkeiten daher gering. Dies stellt aber für die meisten Automotive-Schaltkreise kein Problem dar.

4.1.1 Modelleigenschaften der Ausgangsstufe

Zu Beginn der Modellierung liegen alle Modelleigenschaften, die sowohl vom Designprozess als auch vom virtuellen Test kommen (siehe Tabelle 4.1), zur Verfügung (= Menge E). Diese

Eigenschaften müssen dann nach der Definition in Kapitel 3.2.1 klassifiziert und den Mengen L_1, L_2 und L_3 zugewiesen werden.

	Modelleigenschaften
VT	I_{out} (max und min)
	V_{out} (max und min)
	R_{out}
	Slew-Rate (Anstieg- und Abfallzeit)
	V und I der Bulk-Diode (für Open/Short-Test)
IC-Design	Eigenschaften von VT
	$R_{DS,on}, R_{DS,off}$
	I_{gate}, V_{GS} der Ausgangstransistoren (PMOS, NMOS)
	$t_d, td_{nmos}, td_{pmos}$ Verzögerungszeiten
	Temperaturverhalten

Tabelle 4.1: Modelleigenschaften einer Ausgangsstufe

Klassifizierung der Modelleigenschaften

Alle Modelleigenschaften müssen den Mengen L_1, L_2 und L_3 zugewiesen werden. Die nachfolgende Tabelle (siehe Tabelle 4.2) soll dies verdeutlichen.

Die Mengen L_1, L_2 und L_3 werden den jeweiligen Modell-Level zugewiesen. Der Übergang von den Eigenschaften aus L_1 nach L_2 bedingt eine Topologieänderung des Modells. So wird beispielsweise t_d von L_1 in die Eigenschaften $t_{d,nmos}$ und $t_{d,pmos}$ von L_2 überführt. Ebenso ist es mit der ME R_{out} die mit I_{out} und V_{out} aus L_2 beschrieben wird. Beim Übergang von L_2 nach L_3 werden die Eigenschaften Slew-Rate und I_{out} durch die Eigenschaften aus L_3 mit $I_{gate,nmos}$ und $I_{gate,pmos}$ sowie die Eigenschaften der Ausgangstransistoren z.B. I_{DS} über V_{GS}, ersetzt. Die Definition dieser Vorgehensweise kann in Kapitel 3.2.1 nachgelesen werden.

4.1.2 Level3-Modell einer Ausgangsstufe

Bild 4.1 zeigt einen Low-Side- bzw. High-Side Schalter, der mit einer Mixed-Level Modellierungsmethodik realisiert ist.

Ziel dieser Methode ist es, wie in Kapitel 3.3.1 beschrieben, ein möglichst identisches Verhalten an den Ausgängen nachzubilden und alle Modelleigenschaften von L_3zu berücksichtigen. Dafür sorgen in erster Linie die zwei Transistoren PMOS (MP1) und NMOS (MN1), die sich

Model-Level	**Modelleigenschaften**
Level1	V_{out} (max und min)
	$R_{out} = R_{DS,on}$ und $R_{DS,off}$
	V und I der Bulk-Diode
	Slew-Rate (grobe Nachbildung)
	t_d Verzögerungszeit
Level2	alle ME aus L_1 außer t_d und R_{out}
	Slew-Rate (Anstieg- und Abfallzeit)
	I_{out} (max und min)
	$t_{d,nmos}$, $t_{d,pmos}$ Verzögerungszeiten
Level3	alle ME aus L_2 außer Slew-Rate und I_{out}
	$I_{gate,nmos}$, $I_{gate,pmos}$
	I_{DS} über V_{GS} der Ausgangstransistoren (PMOS, NMOS)
	Temperaturverhalten
	Transistoreigenschaften z.B.Gate-Kapazitäten

Tabelle 4.2: Klassifizierung der Modelleigenschaften

direkt an den Ausgängen befinden und somit die eigentliche Schaltfunktion der Stufe realisieren. Diese Transistoren enthalten auf Transistorebene meist BSIM3, EKV [Buc96] oder einfache Spice Modelle, die durch W und L skalierbar sind.

Um die Anstiegs- und Abfallgeschwindigkeit (Slew-Rate) der Transistoren zu beeinflussen, muss der dafür verantwortliche Gatestrom an den beiden Ausgangstransistoren kontrolliert und nachgebildet werden. Aus diesem Grund werden die vorgeschalteten Buffer-Stufen ebenfalls mit skalierbaren PMOS- und NMOS-Transistoren realisiert. Über W/L lässt sich sowohl der positive als auch der negative Gatestrom regulieren. Die Ansteuerung dieser Buffer-Stufen erfolgt durch eine Delay-Stufe. Diese Stufe beinhaltet, sowohl für das Schalten des PMOS als auch des NMOS Transistors, eine Totzeit „deadtime“ und die Anstieg- und die Abfallzeit „rise/fall-time“. Eine weitere Komponente des Modells ist ein 5 Volt Linearregler, der als Levelshifter eingesetzt wird und für die Ansteuerung von Hochvoltelementen dient. Das digitale Eingangssignal, welches die Ansteuerung der einzelnen Ausgänge vornimmt, wird durch das Kernmodell oder durch die Kontrollogik des Schaltkreises ausgelöst.

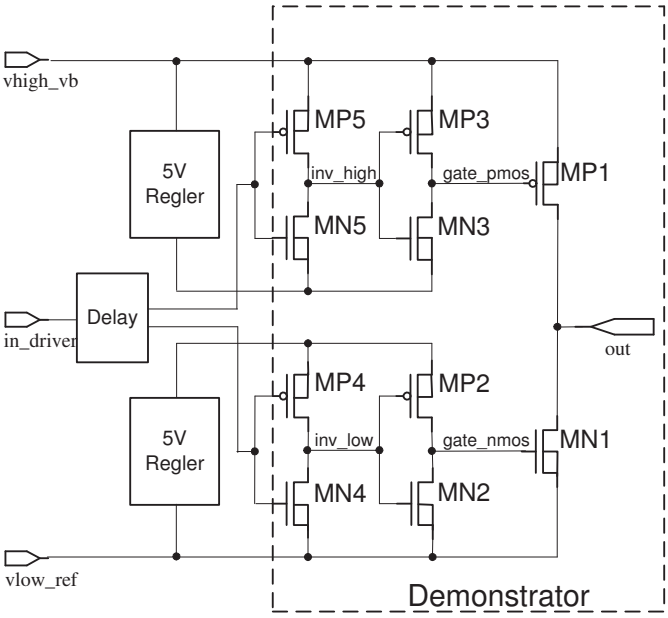

Abbildung 4.1: Level3-Modell einer Ausgangsstufe

4.1.3 Level2-Modell einer Ausgangsstufe

Bei der Vereinfachung des Level3-Modells werden die Buffer-Stufen, die zum Regulieren der Gateströme eingesetzt werden, entfernt. Um die Gateströme weiterhin kontrollieren zu können sind Strombegrenzer einzusetzen. Ein exakter Stromverlauf an den Gates wird hierdurch nicht erzeugt, welcher aber auch nicht zu den Eigenschaften von L_2 gehört. Ein messbarer Performancegewinn stellt sich durch den Wegfall der zwei Buffer-Stufen bzw. der acht Transistoren im Vergleich zum Level3-Modell ein.

Diese Modellstruktur ist in Bild 4.2 dargestellt und wird als Level2-Modell bezeichnet. Die Modelleigenschaft I_{out} aus L_2 könnte in diesem Modell durch eine spannungsgesteuerte Stromquelle realisiert werden. Um den Modellierungsaufwand zu reduzieren, werden an dieser Stelle die Eigenschaften der Ausgangstransistoren ausgenutzt, auch wenn die Modelleigenschaften für das Level2-Modell (siehe Tabelle 4.2) nicht gefordert sind.

Das Verhalten bei der Simulation ist nahezu identisch zur Level3 Simulation. Geringe Abweichungen ergeben sich beim Verhalten der Slew-Rate und beim Temperaturverhalten. Grund hierfür ist die vereinfachte Regulierung des Gatestromes, der durch einen Strombegrenzer realisiert ist. Mit diesem Modell werden alle Eigenschaften aus der Menge L_2 erfüllt.

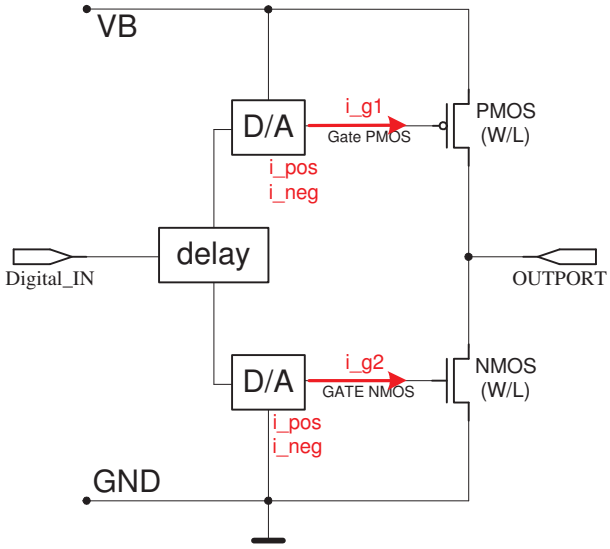

Abbildung 4.2: Level2-Modell einer Ausgangsstufe

4.1.4 Level1-Modell einer Ausgangsstufe

Die Realisierung des Level1-Modells findet auf höchster Abstraktionsebene statt. Es werden nur passive Elemente bei der Modellierung verwendet, die alle durch einer Mixed-Signal Hardware-beschreibungssprache beschrieben werden können (siehe Bild 4.3). Der PMOS und NMOS wird in seiner Grundfunktion durch einen digital umschaltbaren Widerstand realisiert, der je nach Zustand des Digitalsignals R_{out} zwischen $R_{DS,off}$ und $R_{DS,on}$ wechselt.

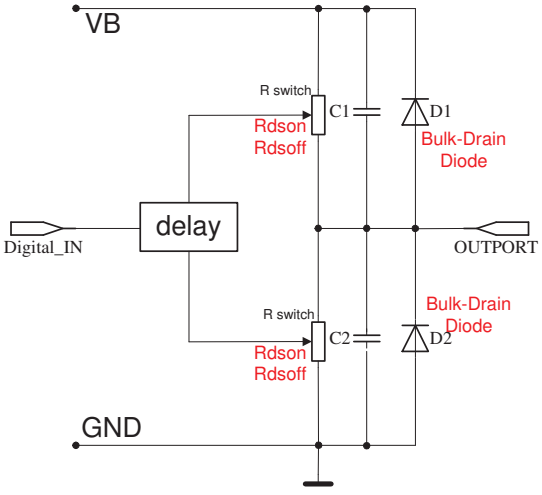

Abbildung 4.3: Level1-Modell einer Ausgangsstufe

Für die Nachbildung der Slew-Rate wird eine Kapazität hinzugefügt, die so dimensioniert werden muss, dass sich ein idealisiertes Verhalten zur originalen Transistorschaltung ergibt. Dies ist mit einer verträglichen Modellabweichung umsetzbar, da das RC-Verhalten meist nicht mit einem Slew-Rate Verhalten eines MOS-Transistors übereinstimmt. Diese Modelleigenschaft ist aber in diesem Fall nicht für das Level1-Modell gefordert. Für die Messtechnik ist in den meisten Fällen nur die Zeit zwischen 10% und 90% des Endsignalzustands von Bedeutung. Diesbezüglich lässt sich das RC-Verhalten so dimensionieren, dass die in der Spezifikation festgelegte zu messende Zeit erreicht wird.

Ein weiteres Element des Modells ist die Diode, welche die Bulk-Drain-Diode des MOS-Transistors nachbildet. Diese Diode wird bei OPEN/Short-und ESD-Messungen benötigt. Bei diesen Testgruppen, wird die Versorgungsspannung auf Masse gelegt. Danach wird zum einen ein positiver und zum anderen ein negativer Strom eingebracht. Als Messergebnis wird eine positive und eine negative Diodenflussspannung erwartet, die durch die Diode im Level1-Modell und durch die Bulk-Drain-Diode im Level2 und Level3-Modell nachgebildet wird. Die Ansteuerung der Widerstände wird von einer digital realisierten Delay-Stufe vorgenommen, die sich in der selben Komplexität wie in den Level2 und Level3-Modellen wiederfindet.

4.1.5 Variantenbildung

Mit dem Ziel ein Bibliothekselement zur Verfügung zu stellen, müssen für die Vielzahl von verschiedenen MOS-Ausgangsstufen Modell-Modifikationen durchgeführt werden. Aus diesem Grund werden die zuvor beschriebenen Modelle (Level1 bis Level3) in verschiedenen Varianten bereitgestellt. Es handelt sich hierbei um Eigenschaften, die eine Topologieänderung zur Folge hat. In Bild 4.4 ist ersichtlich, wie ein Level1-Modell mit verschiedenen Varianten realisiert werden kann. Dabei handelt es sich um folgende Erweiterungen:

- Einfügen von Widerständen (R1, R2, RS) um bei verschiedenen Tests die zu messenden Bedingungen, die sich aus der PV ergeben, zu erfüllen.
- Einfügen von Pull-Up oder Pull-Down Widerständen
- Einfügen von Strom-Limiter

Es gibt Ausgangsstufen, bei denen für das Ein- und Ausschalten mehrere Transistoren in Serie verwendet werden. Dies kann z.B. beim Schalten von hohen Spannungen der Fall sein. Dabei können $R_{DS,off}$ / $R_{DS,on}$ und die Kapazitäten zusammengefasst werden. Die Bulk-Drain-Diode, die beim ESD- und Open/Short-Test benötigt wird, muss so modifiziert werden, dass sich beim Messen die geforderte Spannung einstellt. Wenn beispielsweise drei NMOS in Reihe

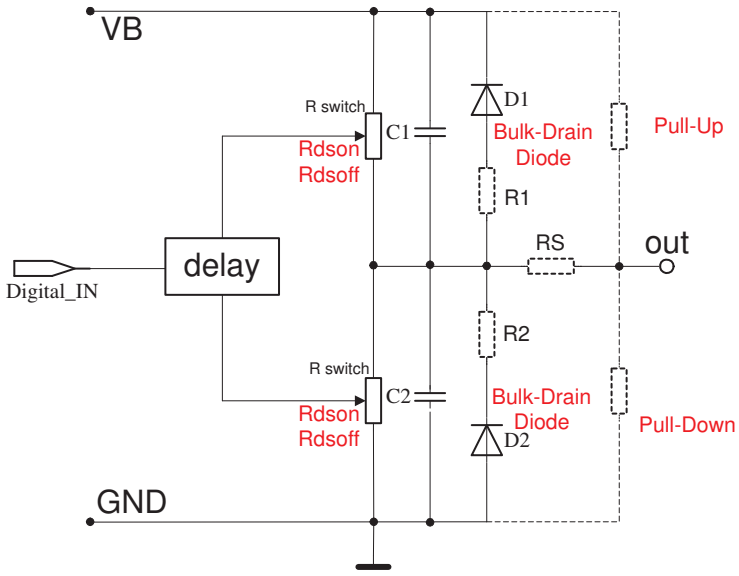

Abbildung 4.4: Level1-Modellvariante einer Ausgangsstufe

verwendet werden, muss die Bulk-Drain-Diode ebenfalls dreimal vorhanden sein, und somit die im Test benötigte Spannung von ca. 2,1 Volt (3*0,7V) liefern. Um die Anzahl der Dioden so gering wie möglich zu halten (aus Performancegründen), kann an dieser Stelle eine Vereinfachung durch einen Widerstand vorgenommen werden. Diese Widerstände (R1, R2, Rs) müssen so dimensioniert sein, dass die im Test geforderten Spannungen, beim Einprägen eines fest in der PV definierten Strom, abfallen. Als weitere Modifikationen können entweder Pull-Up- oder Pull-Down Widerstände eingefügt werden (siehe Bild 4.4).

4.1.6 Simulationsergebnisse

Die Ausgangstreiberstufe aus Bild 4.3, 4.2 und 4.1 wird am Ausgang mit einer ohmschen Last belastet und das Ein- und Ausschalten mit dem AMS-Designer von der Firma Cadence simuliert.

Das Simulationsergebnis in Bild 4.5 zeigt den Vergleich der drei Modelle (Level1, Level2, und Level3), die mit dem gleichen Eingangssignal angesteuert werden. Hieraus ist ersichtlich, dass sich nur das Level1-Modell, bedingt durch die RC-Modellierung, unterschiedlich gegenüber dem Level2 und Level3-Modell verhält. Diese Modelle sind ihrerseits bei Raumtemperatur fast identisch und weisen nur bei verschiedenen Temperaturen Unterschiede auf.

Der Modellfehler wird bei der Anstiegsflanke der Ausgangsspannung sowie bei der Spannung im ein- und ausgeschalteten Zustand ermittelt. Tabelle 4.3 zeigt die Ergebnisse der Modellfehler

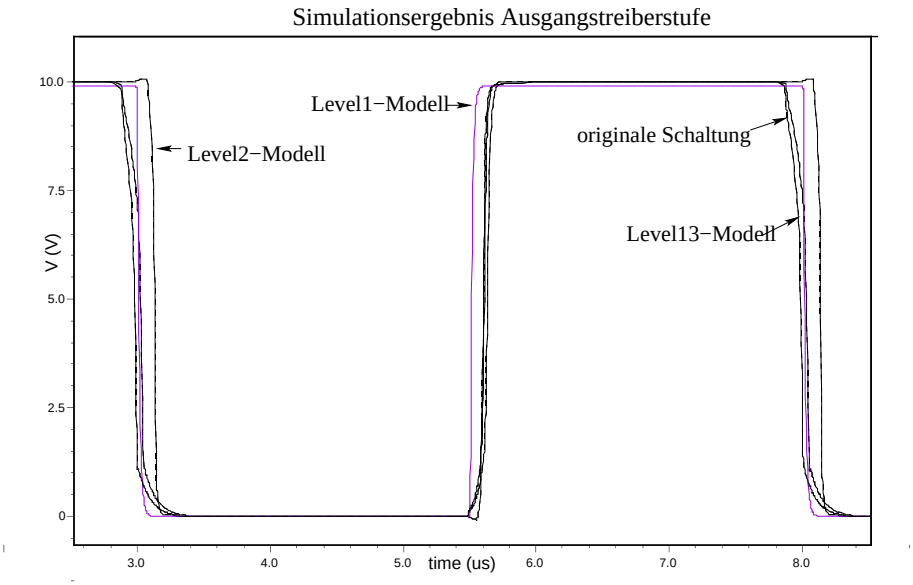

Abbildung 4.5: Simulationsergebnisse verschiedener Modell-Level einer Ausgangsstufe

und den Performancegewinn der einzelnen Modell-Level.

Modell-Level	**Performancegewinn**	**tr Abweichung**
Level1	1660 x	15,9 %
Level2	421 x	9,1 %
Level3	98 x	3,2 %

Tabelle 4.3: Simulationszeit und Modellgenauigkeit bei der Ermittlung der Anstiegszeit

4.2 Realisierung von konfigurierbaren HDL-Modellen

Nachfolgend werden drei Methoden der Realisierung von konfigurierbaren HDL-Modellen, deren Methodik und Herleitung in Kapitel 3.3.3 beschrieben sind, vorgestellt. Als Hardwarebeschreibungssprache wird Verilog-A/AMS verwendet.

Realisierungsmöglichkeiten für konfigurierbare HDL-Modelle in Verilog-AMS:

- mit Präprozessoranweisungen, bei dem Teile des Quelltextes vor seiner Übersetzung ausgeblendet werden,

- mit Parameterübergabe und

- ein dynamisches, das Veränderungen der Modelleigenschaften während der Simulation ermöglicht.

4.2.1 Statisches HDL-Modell mit Präprozessoranweisungen

Durch Verwendung von Präprozessoranweisungen wie *ifdef* können aus dem Quelltext des Modells beliebige Bereiche ausgewählt werden. Vorteilhaft an dieser Realisierung ist, dass explizit nur diese Bereiche ins kompilierte Modell übernommen werden. Nachteil ist, dass die Auswahl vor dem Kompilieren getroffen werden muss und diese Einstellung global für alle Instanzen des HDL-Modells übernommen wird. Im folgenden Quelltext wird dieser Ansatz schematisch dargestellt:

```
‘ifdef res_nodal
      - Funktion abc
‘else
      - keine Funktion abc
‘endif
```

Ein weiterer Vorteil des Ansatzes ist die Möglichkeit der Veränderung der Schaltungstopologie durch Hinzufügen oder Weglassen interner Knoten. Am Beispiel der Bahnwiderstände von MOSFET HDL-Modell wird dies deutlich: Sie können nur unter Verwendung interner Knoten ins Modell aufgenommen werden. Nur mit Hilfe von Präprozessoranweisungen kann ihre Deklaration (*electrical Bi, Di, Si, Gi;*) wahlweise verwendet werden. Im weiteren Quelltext wird dann zwischen internen Knoten (Si, Di etc.) und den Pins (D, S etc.) gewechselt:

```
‘ifdef res_nodal
    vds = V(Di, Si);
    ...
‘else
    vds = V(D, S);
   ...
‘endif
```

4.2.2 Statisches HDL-Modell mit Parameterübergabe

Beim statischen HDL-Modell mit Parameterübergabe wird anstelle von Präprozessoranweisungen das *if-else* -Konstrukt verwendet. Das Aus- und Einschalten der einzelnen Funktionen wird durch Parameterübergabe gesteuert und kann für jede Modellinstanz individuell durchgeführt werden. Dies ist ein Vorteil gegenüber der Verwendung von Präprozessoranweisungen. Als Nachteil zeigt sich ein etwas höherer Rechenaufwand bei der Simulation.

```
parameter integer res_nodal = 0;
.....
if (res_nodal == 1)
```

```
      - Funktion Bahnwiderstände
else
      - keine Funktion Bahnwiderstände
end
```

4.2.3 Dynamische HDL-Modelle

Einige Anwendungen wie z. B. der Virtuelle Test, erfordern, dass eine ganze Testgruppe aus mehreren Tests zusammen simuliert wird. Somit ergibt sich während der Simulation die Forderung nach dynamisch umschaltbaren HDL-Modellen. Ermöglicht wird dies durch Verwendung von Variablen anstelle der in Abschnitt 4.2.2 beschriebenen Parameter.

```
integer res_nodal;
...
if (res_nodal == 1)
      - Funktion Bahnwiderstände
else
      - keine Funktion Bahnwiderstände
end
```

Das Umschalten dieser Variablen kann in einer Testbench instanzspezifisch erfolgen. Im nachfolgenden Quelltext ist ein Beispiel einer Testbench dargestellt, in der die Bahnwiderstände einer Instanz nach 10.000 Zeiteinheiten hinzugeschaltet werden.

```
initial begin
   I_top.I_inv.I_NMOS1.res_nodal = 0;
   # 10000
   I_top.I_inv.I_NMOS1.res_nodal = 1;
......
end
```

Zur Zeit lässt der AMS-Designer und Spectre analoge Anweisungen [LRM04] in Verilog-A innerhalb von *if*-Konstrukten mit Variablen nicht zu. Daher müssen diese Anweisungen, z.B. die Berechnung der Kapazitäten mit *ddt* oder *limexp* für die Diodenberechnung, außerhalb der *if*-Konstrukte liegen. Dadurch wird der mögliche Rechenzeitgewinn bei der Simulation mit variablen dynamischen Modellen nicht erreicht. Im folgenden Abschnitt werden deshalb statische Modelle mit Parameterübergabe verwendet.

4.3 Konfigurierbares MOSFET HDL-Modell

In diesem Abschnitt wird das konfigurierbare MOSFET HDL-Modelle am Beispiel des „MOS-Level1-Modells“ vorgestellt. Die Effekte, die das vollständige MOSFET HDL-Modell umfasst,

können im konfigurierbaren Modell gezielt eingeschaltet oder ausgeblendet werden. Dies ermöglicht es, nur die für das Verhaltensmodell benötigten Effekte auszuwählen und damit die Simulationszeiten mit geringem Aufwand erheblich zu senken. Weiter wird am Beispiel des „MOS-Level1-Modells“ die Partitionierung nach Effekten und die Implementierung eines konfigurierbaren HDL-Modells in Verilog-A aufgezeigt. Es werden Anwendungsbeispiele dargestellt, die den Simulationszeitgewinn demonstrieren. Die Methode der konfigurierbaren MOS-Modelle ist grundsätzlich mit allen Transistormodellen anwendbar und in Schaltungssimulatoren integrierbar. Ein weiteres Ziel ist es, MOSFET HDL-Modelle zu generieren, die universell für mehrere Anwendungsbereiche der Modellierung, z. B. im Entwurfsprozess (Top-Down-Methodik) oder dem virtuellen Test, einsetzbar sind.

Zur Demonstration dieser Methode wird eine Ausgangstreiberstufe, die mit einer Mixed-Level Modellierung realisiert ist, herangezogen. Hierbei sollen nur die für diese Modellierungstechnik relevanten Transistoren untersucht und als Demonstrator verwendet werden (siehe Bild 4.1). In der Praxis werden bei der Schaltungsentwicklung nur noch selten MOSFET-Modelle niedrigen Levels eingesetzt. Hier haben sich in den meisten Fällen z.B. schon BSIM oder EKV etabliert. So auch bei dem in diesem Beitrag vorgestellten Demonstrator, dessen originale Transistormodelle auf Basis von EKV realisiert sind. Auf eine nähere Erläuterung der Methode mit EKV-Modellen soll wegen der Komplexität der Berechnungen hier verzichtet werden, statt dessen werden „Level1-Modelle“ verwendet. Aus diesem Grund werden alle diese komplexen Transistormodelle in „Level1-Modelle“ überführt, die in den meisten Fällen ausreichend für die Anforderungen an das Verhaltensmodell sind [Web07B]. Dieses MOSFET-Verhaltensmodell ist in Verilog-A realisiert und wird im Spectre Simulator von Cadence eingesetzt.

4.3.1 Realisierung in Verilog-A

Dieser Abschnitt beschreibt, wie auf der Basis des „Level1-Modells“ ein konfigurierbares Verilog-A-Verhaltensmodell erstellt wird. Ziel bei der Entwicklung des Modells ist es, die benötigten Modelleffekte auszuwählen und die übrigen aus dem Modell ausblenden zu können. Damit wird bei der Simulation eine minimale Jacobi-Matrix erreicht und der Rechenaufwand reduziert. Hierfür werden ausgehend vom „Level1-Modell“ die Berechnung des Drainstroms, des Substrateffekts sowie die Einbeziehung der Kapazitäten, Widerstände und Dioden, variabel gemacht. Tabelle 4.4 führt die unterschiedlichen Varianten bei der Berechnung des Drainstroms auf. Ferner können die in Tabelle 4.5 dargestellten Bestandteile des MOSFET-Verhaltensmodells ausgewählt werden.

Variante	Beschreibung
0	Spannungsgesteuerter Widerstand
1	Ohne Kanallängenmodulation
2	Mit Kanallängenmodulation
3	Mit Kurzkanaleffekt

Tabelle 4.4: Berechnung des Drainstroms im Verilog-A-Modell

Bestandteil	Beschreibung
res_nodal	Bahnwiderstände
cap_gate	Gate-Kapazitäten
cap_sub	Sperrschichtkapazitäten
dio_sub	Substratdioden
threshold	Substrateffekt

Tabelle 4.5: Wählbare Bestandteile des MOSFET Verilog-A-Modells

Arbeitsbereichserkennung

Bei der Erkennung in welchem Arbeitsbereich sich der Transistor befindet, wird die Variable *region* verwendet:

```
if ((vgs <= vtho)  || (vds <= 0))
    region = 1;
else if ((vds < (vgs-vtho)) && (vgs>vtho))
    region = 2;
else
    region = 3;
```

Berechnung des Drainstroms

In dem Verhaltensmodell gibt es vier Methoden den Drainstrom zu berechnen (siehe Tabelle 4.4). Bei der einfachsten Variante ($var = 0$) wird nicht zwischen linearem und Sättigungsbereich unterschieden. Der Drainstrom wird durch $I_D \approx \beta \cdot ((U_{GS} - U_{th}) \cdot U_{DS})$ berechnet. Dies entspricht einem spannungsgesteuerten Widerstand mit

$$R_{DS} = \frac{1}{\beta \cdot (U_{GS} - U_{TH})}, \quad \beta = \frac{K_n \cdot W}{L} \tag{4.1}$$

Eine weitere Variante ($var = 1$) ist die Berechnung basierend auf der Annahme einer idealen Ladungsverteilung im Kanal (Sah's Modell). Die Kanallängenmodulation wird mit ($var = 2$) berücksichtigt und nach Gleichung 2.1 berechnet (Shichman-Hodges-Modell). Mit dem Parameter

λ wird dieser Effekt gesteuert. Bei der letzten Variante ($var = 3$) wird zusätzlich der Kurzkanaleffekt wie in Gleichung 2.2 dargestellt berücksichtigt.

Im nachfolgenden Quelltext ist die Realisierung der Varianten der Drainstromberechnung in Verilog-A dargestellt:

```
if (var == 0) begin
  case(region)
    1: id = ‘ids;
    2: id = beta * (vgs - vtho)  * vds;
    3: id = beta * (vgs - vtho)  * vds;
    default: id = ‘ids;
  endcase
end
//-----------------------------------------------/
if (var == 1) begin
 case(region)
  1: id = ‘ids;
  2: id = beta*((vgs-vtho)-(vds/2))*vds;
  3: id = (beta/2)*(pow((vgs-vtho),2));
  default: id =‘ids;
 endcase
end
//-----------------------------------------------/
if (var == 2) begin
early_effect = 1 + lambda * vds;
 case(region)
  1: id = ‘ids;
  2: id = beta*((vgs-vtho-(vds/2))*vds*early_effect;
  3: id = (beta/2)*(pow((vgs-vtho),2))*early_effect;
  default: id = ‘ids;
 endcase
end
//-----------------------------------------------/
if (var == 3) begin
 case(region)
  1: id = ‘ids;
  2: id = beta * ((vgs-vtho)-(vds/2))*vds
       *(1+lambda*((1/(2e5*Leff))+1)*vds);
  3: id = (beta / 2)*(pow((vgs-vtho), 2))
       *(1+lambda*((1/(2e5*Leff))+1)*vds);
  default: id = ‘ids;
  endcase
end
```

Stromzuweisung

Abhängig davon, ob die Bahnwiderstände im Modell verwendet werden, wird der Drainstrom den internen Klemmen *Di* und *Si* oder den externen Klemmen *D* und *S* zugewiesen.

```
‘ifdef res_nodal
   I(Di,Si)<+ id;
‘else
   I(D,S)<+ id;
‘endif
```

Bahnwiderstände

Die Bahnwiderstände befinden sich zwischen den externen und den internen Knoten. Bei der Modellierung werden die effektiven Widerstände an Drain und Source (R_D und R_S) verwendet. Keine Berücksichtigung finden die Widerstände an Bulk und Gate.

```
‘ifdef res_nodal
   V(S, Si) <+ I(S, Si) * RS;
   V(D, Di) <+ I(D, Di) * RD;
‘endif
```

Gate-Kapazitäten

Die Gate-Kapazitäten werden für die drei Arbeitsbereiche in Abhängigkeit von C_{ox} und den anliegenden Spannungen unterschiedlich berechnet [Che99]. Die Überlappungskapazitäten werden hier ebenfalls berücksichtigt.

Ein Problem bei der Simulation stellt das Schalten der Kapazitäten zwischen den verschiedenen Bereichen dar. Als Lösung wird hierfür die Anweisung *transition* verwendet, die ein sanftes Umschalten ermöglicht.

```
‘ifdef cap_gate
  if (region == 1) begin
   cgsk=0;
   cgdk=0;
   cgbk=cox;
  end
  if (region == 2) begin
   cgsk=((2*cox)/3)*(1-pow(((vgs-vtho-vds)
        / (2*(vgs-vtho)-vds)),2));
   cgdk=((2*cox)/3)*(1-pow(((vgs-vtho)
        / (2*(vgs-vtho)-vds)),2));
   cgbk=0;
  end
```

```
  if (region == 3) begin
   cgsk=(2*cox)/3;
   cgdk=0;
   cgbk=0;
  end
qgs = (transition(cgsk)+ ‘cgso*W)* vgs;
qgd = (transition(cgdk)+ ‘cgdo*W)* vgd;
qgb = (transition(cgbk)+ ‘cgbo*L)* vgb;
  ‘ifdef res_nodal
     I(Gi,Di) <+ ddt(qgd);
     I(Gi,Si) <+ ddt(qgs);
     I(Gi,Bi) <+ ddt(qgb);
   ‘else
     I(G,D) <+ ddt(qgd);
     I(G,S) <+ ddt(qgs);
     I(G,B) <+ ddt(qgb);
   ‘endif
‘endif
```

Sperrschichtkapazitäten

Für die Sperrschichtkapazitäten sind die pn-Übergänge zwischen Substrat und Source sowie Substrat und Drain verantwortlich. Diese Kapazitäten sind spannungsabhängig (siehe [Che99]).

```
‘ifdef cap_sub
  fbp =‘FC*‘mj;
  if (vbd <= fbp)
   cbd=‘cj*‘Abd * (1-(vbd*1/‘PB));
  else
   cbd=((‘cj*‘Abd)/pow((1-‘FC),1+‘mj))
      *(1-(1+‘mj)*‘FC+‘mj*vbd/‘PB);
  if (vbs <= fbp)
   cbs =‘cj*‘Abs*(1-(vbs*1/‘PB));
  else
   cbs=((‘cj*‘Abs)/pow((1-‘FC),1+‘mj))
*(1-(1+‘mj)*‘FC+‘mj* vbs/‘PB);

  ‘ifdef res_nodal
     I(Bi,Si) <+ ddt(cbs * vbs);
     I(Bi,Di) <+ ddt(cbs * vbs);
  ‘else
     I(B,S) <+ ddt(cbs * vbs);
     I(B,D) <+ ddt(cbs * vbs);
```

```
   ‘endif
‘endif
```

Substratdioden

Die Substratdioden liegen zwischen den internen Knoten von Bulk und Drain bzw. Source. Mit *vt* wird die Temperaturspannung, die im Simulator errechnet wird, verwendet. Der Sättigungssperrstrom ‘*is* wird sowohl für $I_{S,S}$ als auch für $I_{S,D}$ verwendet.

```
‘ifdef  dio_sub
   ibd = ‘is*(limexp(vbd /$vt)-1.0);
   ibs = ‘is*(limexp(vbs /$vt)-1.0);
   ‘ifdef res_nodal
      I(Bi,Di) <+ ibd ;
      I(Bi,Si) <+ ibs ;
   ‘else
      I(B,D) <+ ibd ;
      I(B,S) <+ ibs ;
   ‘endif
‘endif
```

Substrateffekt

Die Schwellspannung ist hauptsächlich von der Bulk-Source-Spannung abhängig. Als Modellparameter wird die Nullschwellspannung *vt0* übergeben.

```
‘ifdef  threshold
  vtho = vt0+(gamma*((sqrt(abs(phi-vbs)))-(sqrt(phi))));
‘else
  vtho = vt0;
‘endif
```

4.3.2 Anwendungsbeispiele

In diesem Abschnitt wird der Einsatz konfigurierbarer MOSFET-Verhaltensmodelle am Beispiel einer Ausgangstreiberstufe demonstriert, die Mixed-Level modelliert ist. Ferner wird auf die Ergebnisse der Einzeltransistoren sowie der Treiberstufe eingegangen und abschließend eine Auswertung dargestellt. Bei dem hier verwendeten Beispiel handelt es sich um die Endstufe eines Ausgangstreibers, die in Bild 4.6 dargestellt ist. Untersucht werden nur die Transistoren, die bei einer Mixed-Level Modellierung dieser Treiberstufe benötigt werden (MP1, MP2, MP3, MP4, MP5, MN1, MN2, MN3, MN4, MN5). Die verbleibenden Komponenten, die zusätzlich

zur Mixed-Level Modellierung der Stufe verwendet wurden, werden in diesem Anwendungsbeispiel nicht betrachtet. Die EKV-Transistormodellparameter, der zu diesem Beispiel gehörenden Technologie, wurden in Parameter für das „Level1-Modell“ überführt (siehe hierzu D.2).

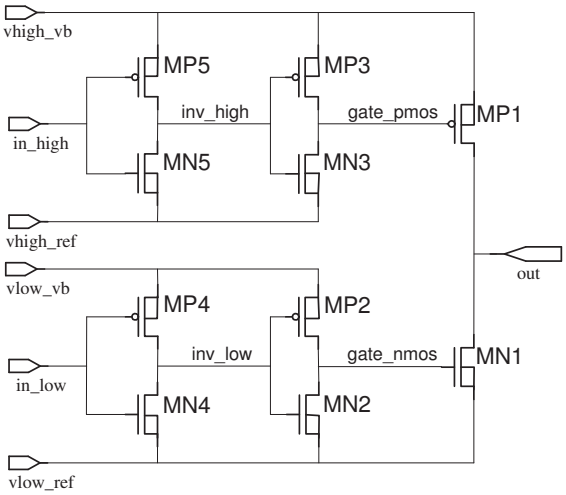

Abbildung 4.6: Endstufe eines Ausgangstreibers

DC-Verhalten mit variablem MOSFET-Modell

Bild 4.7 zeigt die simulierte Ausgangskennlinie des NMOS mit verschiedenen Einstellungen des variablen MOSFET-Verhaltensmodells. Die Kapazitäten (Gate- und Sperrschichtkapazitäten) haben keinen Einfluss auf das DC-Verhalten. Gleiches gilt für den Substrateffekt, da Bulk und Source verbunden sind.

Anwendung des konfigurierbaren MOSFET-Modells

Folgende Aspekte müssen bei der Konfiguration des MOSFET-Verhaltensmodells berücksichtigt werden:

- Applikation der Schaltung (Testbench)
- Arbeitspunkt der Schaltung bzw. des Transistors
- Simulationsart (DC, transient etc.)
- Geforderte Genauigkeit der Simulation
- Zweck der Simulation (virtueller Test, Entwicklungsphase, Systemsimulation etc.)

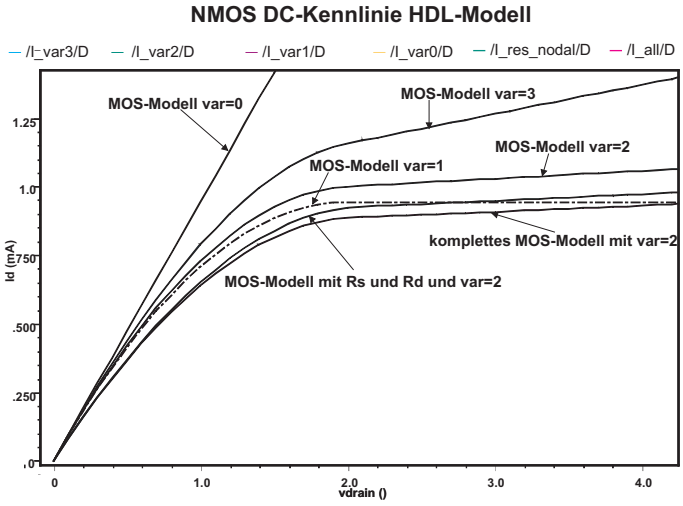

Abbildung 4.7: MOSFET-Kennlinie für verschiedene Einstellungen des variablen MOSFET-HDLs

Zur Auswahl der richtigen Einstellungen ist schaltungstechnisches Wissen erforderlich. Die Einstellungen erfolgen direkt am Transistorsymbol im Schaltplaneditor (siehe Bild 4.8). Die fol-

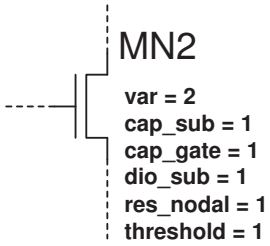

Abbildung 4.8: Konfiguration des Verhaltensmodells am Symbol

genden Beispiele zeigen das Vorgehen bei verschiedenen Simulationsaufgaben.

Ermittlung von $R_{DS,on}$

Bei den meisten Treiberstufen wird beim Produktionsmessen der Parameter $R_{DS,on}$ überwacht. Dieses Verhalten soll im virtuellen Test durch das Verhaltensmodell wiedergeben werden. Bei der Ermittlung von $R_{DS,on}$ wird am Ausgang ein Strom von 40 mA als Last angelegt. Danach wird der Spannungsabfall vom Ausgang zur Betriebsspannung (10 V) gemessen und der Widerstandswert berechnet. Für diese Anforderung wurde die in Tabelle 4.6 aufgeführte Transistor-Konfiguration der Treiberstufe eingestellt.

Variante	Mx1	Mx2, Mx3	Mx4, Mx5
var	1	2	2
res_nodal	ja	nein	nein
cap_gate	ja	nein	nein
cap_sub	ja	nein	nein
dio_sub	ja	nein	nein
threshold	nein	nein	nein

Tabelle 4.6: Konfigurationen zur Bestimmung von $R_{DS,on}$

Die beiden Ausgangstransistoren MN1 und MP1 beinhalten alle Eigenschaften außer dem Substrateffekt, da Bulk und Source kurzgeschlossen sind. Bei den ansteuernden Transistoren sind alle Eigenschaften deaktiviert worden. Das Simulationsergebnis in Bild 4.9 zeigt die Ausgangsspannung für drei verschiedene Einstellungen der MOSFET HDL-Modelle: mit allen Eigenschaften (komplette MOSFET-Modelle), mit Abwahl aller Eigenschaften und $var = 0$ (spannungsgesteuerter Widerstand) sowie mit den in Tabelle 4.6 dargestellten Einstellungen. Durch die Vereinfachung bei den ansteuernden Transistoren weichen die Verzögerungen sehr stark gegenüber dem vollständigen Modell ab. Die Endwerte, bei denen die Ermittlung des Wertes $R_{DS,on}$ stattfindet, sind bei den Einstellungen nach Tabelle 4.6 identisch, mit denen des vollständigen Modells.

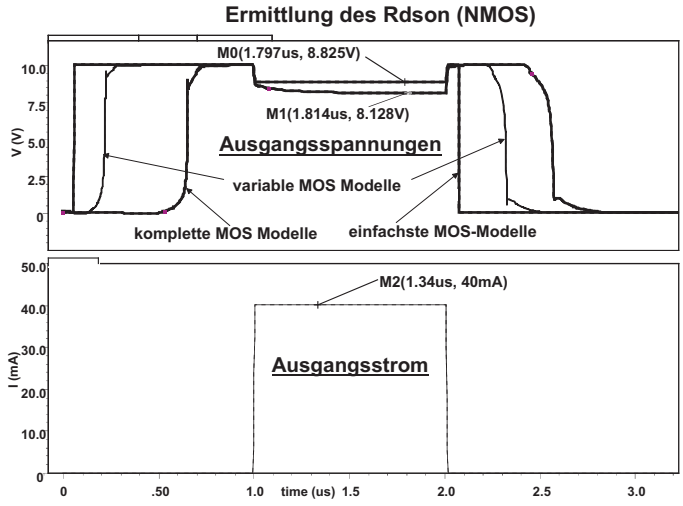

Abbildung 4.9: Ermittlung des $R_{DS,on}$(NMOS) der Treiberstufe

Die Simulation der Treiberstufe mit den nach Tabelle 4.6 eingestellten Parametern ist 6,7 x schneller als bei der Verwendung der MOSFET-Modelle mit allen Eigenschaften.

Dass Transistormodelle, die als Primitive im Simulator implementiert sind, optimale Performance bieten, ist hinlänglich bekannt. In diesem Beispiel ist bei der Verwendung der ursprünglichen EKV-Modelle gegenüber den vollständigen Verhaltensmodellen eine 2 x schnellere Simulation erreicht worden. Werden die Zeiten der EKV-Modelle mit den optimal eingestellten MOSFET-Verhaltensmodellen verglichen, wird immer noch eine Beschleunigung um den Faktor drei erreicht. Bei Verwendung der einfachsten Einstellung der MOSFET-Verhaltensmodelle sogar bis zu 22 x, bei einem Fehler von 8,6 %. Tabelle 4.7 fasst diese Ergebnisse zusammen.

Eigenschaften	**Performance**	**Abweichung**
vollständig	1 x	keine
wie in Tab. 4.6	6,7 x	keine
all deaktiviert	22 x	8,6 %

Tabelle 4.7: Simulationsperformance und Modellgenauigkeit bei der Bestimmung von $R_{DS,on}$

Ermittlung der Bulk-Dioden-Spannung

Beim Kontakttest wird ein Strom an dem zu messenden Pin eingespeist und eine Spannung, die der Bulk-Dioden-Spannung entspricht, erwartet. In der Testbench ist hierfür eine Stromquelle am Ausgang angeschlossen, die einmal einen positiven und einmal einen negativen Strom einprägt. Alle anderen Eingänge liegen auf Masse. Die in Tabelle 4.8 aufgeführten Funktionen sind bei den einzelnen Transistoren der Treiberstufe für diese Simulationsaufgabe eingestellt worden. Bei dieser Konfiguration werden nur die Bulk-Dioden und die Bahnwiderstände der Ausgangstransistoren verwendet.

Variante	**Mx1**	**Mx2, Mx3**	**Mx4, Mx5**
var	1	2	2
res_nodal	ja	nein	nein
cap_gate	nein	nein	nein
cap_sub	nein	nein	nein
dio_sub	ja	nein	nein
threshold	nein	nein	nein

Tabelle 4.8: Verwendete Funktionen für die MOSFET-Modelle zur Bestimmung der Bulk-Dioden-Spannung

In Bild 4.10 ist das Simulationsergebnis der Bulk-Dioden-Spannungen dargestellt. Zu erkennen sind die zwei Ausgangsspannungen und der Ausgangsstrom von 1 mA. Tabelle 4.9 führt

Simulationszeit und -genauigkeit auf.

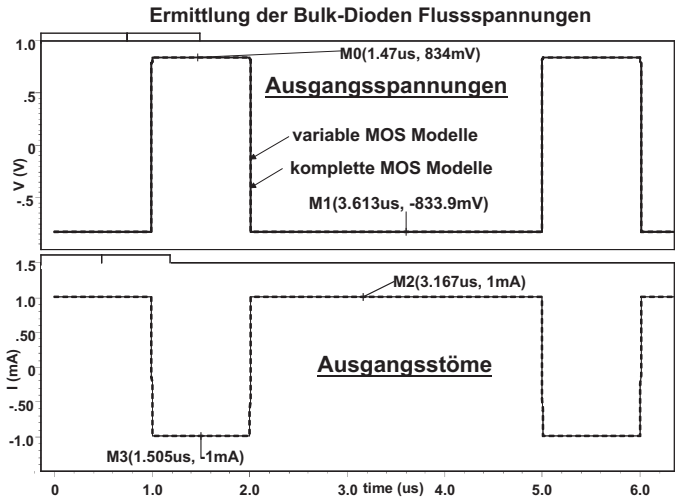

Abbildung 4.10: Ermittlung der Bulk-Dioden-Flussspannung

Performancegewinn	**Modellabweichung**
2,3 x	keine

Tabelle 4.9: Simulationszeit und Modellgenauigkeit bei der Bestimmung der Bulk-Dioden Spannung

Ermittlung der Slew-Rate

Bei der Ermittlung der Slew-Rate wird der Ausgang mit einer ohmschen Last belastet. Die Ermittlung der Anstiegszeit wird bei 2 V Ausgangsspannung (20 % des Endwertes) und 8 V (80 % des Endwertes) gemessen.

Die einzelnen MOSFET-Modelle sind wie bei der Ermittlung des $R_{DS,on}$ eingestellt, siehe Tabelle 4.6. In Bild 4.11 ist das Simulationsergebnis dargestellt.

Hier ist die Simulationsgeschwindigkeit 34 x höher bei einem Fehler von 47 % bei der Anstiegszeit und 15,1 % bei der Abfallzeit, siehe Tabelle 4.10. Falls dieser Fehler nicht akzeptabel ist, müssen ausgeblendete Eigenschaften der ansteuernden Transistoren wieder eingeschaltet werden. Das würde den Performancegewinn reduzieren.

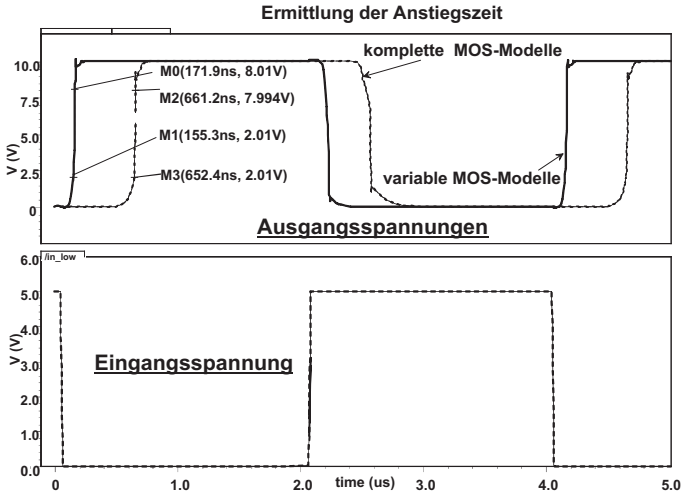

Abbildung 4.11: Ermittlung der Anstiegszeit

Performance	**Abweichung (rise)**	**Abweichung (fall)**
34 x	47 %	15,1 %

Tabelle 4.10: Simulationszeit und Modellgenauigkeit bei der Ermittlung der Anstiegs- und Abfallzeit

Ermittlung der Spannung im ein- und ausgeschalteten Zustand

Als Last wird bei der Ermittlung der Spannungen im ein- und ausgeschalteten Zustand ein Widerstand verwendet. Gemessen werden diese Spannungen immer im eingeschwungenen Zustand, d.h. weit weg von den Anstiegs- und Abfallflanken. Bei korrekter Funktion des Treibers arbeitet der Ausgangstransistor dann im linearen Bereich. Mit diesen Randbedingungen können für alle Transistoren die Verhaltensmodelle mit $var = 0$ und ohne zusätzliche Modelleffekte verwendet werden.

Bild 4.12 zeigt das Simulationsergebnis. Es sind die Ausgangsspannungsverläufe mit den Spannungswerten im ein- bzw. ausgeschalteten Zustand dargestellt. An den Simulationszeiten in Tabelle 4.11 wird der Vorteil des variablen MOSFET HDL-Modells in diesem Beispiel besonders deutlich. Die Simulation ist 217,5 x schneller bei nur 0,15 % Fehler.

Performancegewinn	**Modellabweichung**
217,5 x	0,15 %

Tabelle 4.11: Simulationszeit und Modellgenauigkeit bei der Spannung im ein- und ausgeschalteten Zustand

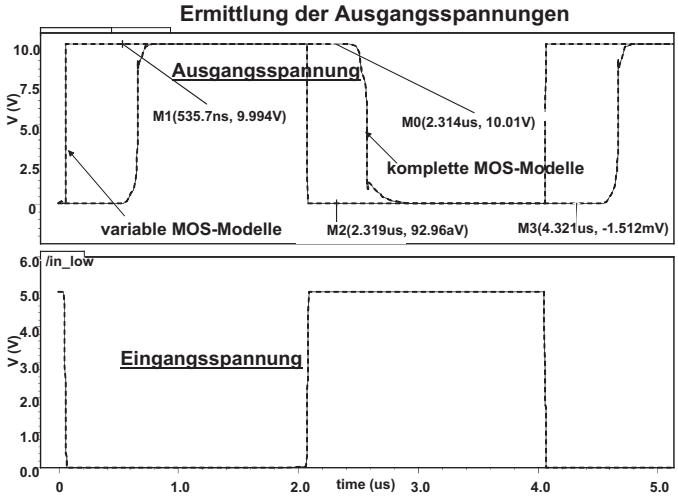

Abbildung 4.12: Ermittlung der Spannung im ein- und ausgeschalteten Zustand

4.3.3 Zusammenfassung

In diesem Abschnitt wurde ein Verfahren vorgestellt, wie MOSFET-Modelle partitioniert und deren Eigenschaften aktiviert oder deaktiviert werden können, mit dem Ziel, einen Performancegewinn zu erhalten. Diese Methode wurde anhand von MOSFET-Verhaltensmodellen, die auf Basis von „Level1-MOSFET“ Berechnungen realisiert wurden, demonstriert. Die Modelle bieten zusätzlich noch auswählbare Erweiterungen wie den Kurzkanaleffekt oder Vereinfachungen, die es erlauben, den Transistor als spannungsgesteuerten Widerstand einzusetzen. Die einzelnen Modelleffekte, die variabel ein- und ausschaltbar sind, wurden einzeln durch den Source-Code beschrieben. Das Modell wurde in Verilog-A erstellt und mit Spectre von Cadence simuliert. Abschließend wurden die konfigurierbaren MOSFET HDL-Modelle an einem Demonstrator angewandt. Hierbei wurde der Performancegewinn und der Modellfehler an vier verschiedenen Simulationsaufgaben ermittelt. Es ergab sich eine Geschwindigkeitszunahme von bis zu 217 x mit einem Fehler von nur 0,15%.

Ein Nachteil zeigte sich bei mehrmals vorkommenden Konvergenzproblemen, die aber immer wieder durch neue Simulatoreinstellungen behoben werden konnten. Die Tatsache, dass die Verwendung der originalen EKV-Transistoren schneller als die komplexeste Ausbaustufe der MOSFET HDL Modelle ist, ist hinreichend bekannt. Um dem entgegenzuwirken, hat Cadence bei den neueren Simulatorversionen einen C-Compiler für Verilog-A implementiert. Trotzdem lässt sich feststellen, dass auch ein Performancegewinn der optimal eingestellten MOSFET HDL-Modelle gegenüber den originalen EKV-Transistoren vorliegt. Um eine Verbesserung der Performance zu erzielen, wäre es wünschenswert, die hier beschriebene Methode in den Simulator zu

integrieren. In diesem Beitrag wurde ein MOS-HDL-Modell auf Basis des Level1-Modell realisiert, dies aber nur als Demonstration der Methode dienen sollte. Sinnvoll wäre es, bei allen MOSFET-Modelltypen eine solche Methode anzuwenden. Das optimale Einstellen der verschiedenen Eigenschaften des Modells erfordert schaltungstechnisches Wissen, welches bei der Modellierung Voraussetzung sein sollte.

4.4 Realisierung von Bibliothekselementen

Die in Kapitel 3.3.2 demonstrierte Methode zur Modellierung von Bibliothekselementen ebenso wie das Verwenden von konfigurierbaren MOSFET-Modellen aus Kapitel 4.3, sollen in diesem Abschnitt kombiniert werden. Ziel ist es, einen noch besseren Performancegewinn versus Modellgenauigkeit zu ermöglichen, um somit ein optimales Verhaltensmodell für definierte Simulationsaufgabe zu entwickeln.

Die konfigurierbaren MOSFET HDL-Modelle lassen sich bei Modellen die mit einer Mixed-Level Modellierung realisiert sind und Transistoren beinhalten, einsetzen. In unserem Beispiel wäre das Level3 und Level2 der Ausgangstreiberstufe (siehe Kapitel 4.1).

4.4.1 Simulationsergebnisse

Zur Demonstration wird nachfolgend das Level3 und das Level2-Modell verwendet. Als Simulationsaufgaben wird, wie in Kapitel zuvor beschrieben, $R_{DS,on}$ ermittelt. Tabelle 4.12 zeigt den Performancegewinn bei dem Level3-Modell der Ausgangsstufe. Als Referenz wird die originale Schaltung herangezogen.

Modell	**Performancegewinn**	**Modellabweichung**
Level1	1660 x	< 1%
Level2	421 x	< 1%
Level3	98 x	< 1%

Tabelle 4.12: Performancegewinn bei der Aktivierung aller Modelleigenschaften im Level-Modell

Bei den Ergebnissen in Tabelle 4.13 sind die Eigenschaften der MOSFET-Modelle so wie in Kapitel 4.3.2 beschrieben, optimal für die jeweilige Simulationsaufgabe eingestellt.

Die Kombination der beiden Methoden ermöglicht es, eine optimale Konfiguration der Verhaltensmodelle für die jeweilige Anforderung einzustellen. Wie aus den Ergebnissen ersichtlich, kann bei der Vereinigung der Methode der Performancegewinn nochmals gesteigert werden.

Modell	Performancegewinn	Modellabweichung
Level1	1660 x	< 1%
Level2	1250 x	< 1%
Level3	454 x	< 1%

Tabelle 4.13: Simulationsperformance und Modellgenauigkeit beim Level Modell

4.5 Bestimmung der Modellgenauigkeit

Die Bestimmung des Modellfehlers kann unterschiedlich erfolgen. Der in den Kapiteln zuvor berechnete Fehler wurde speziell für Messungen im virtuellen Test angewandt. Hier wird in der Messtechnik und somit auch in der Prüfvorschrift genau beschrieben, wann gemessen werden muss. Die Bestimmung der Anstiegszeit bei 10% und 90% des Endwertes ist ein typisches Beispiel.

Eine andere Methode bei der Bestimmung der Modellabweichung wird beispielsweise im Designprozess verlangt. Hier ist meist der Kurvenverlauf in einem bestimmten Zeitabschnitt von Bedeutung. Die hier geforderten Bewertungsmethoden sind in Kapitel E beschrieben. Das Auswerteprogramm SEAP, das in Kapitel E.2 vorgestellt wird, kommt hier zum Einsatz.

Bei der Bestimmung der Modellabweichung der Ausgangstreiberstufe wird in dem Programm, die „minimale Distanz“ als Abstandsmaß und die Gleichung E.10 als Fehlernorm, ausgewählt. In Bild 4.13 ist beispielhaft die Ermittlung der Modellabweichung bei dem Ausgangsstrom der verschiedenen Modell-Level dargestellt.

Je nach Simulationsaufgabe ist der Modellfehler für das gleiche Modell unterschiedlich. Eine pauschale Bewertung der einzelnen Modell-Level bzw. Modellvarianten ist somit fast nicht möglich.

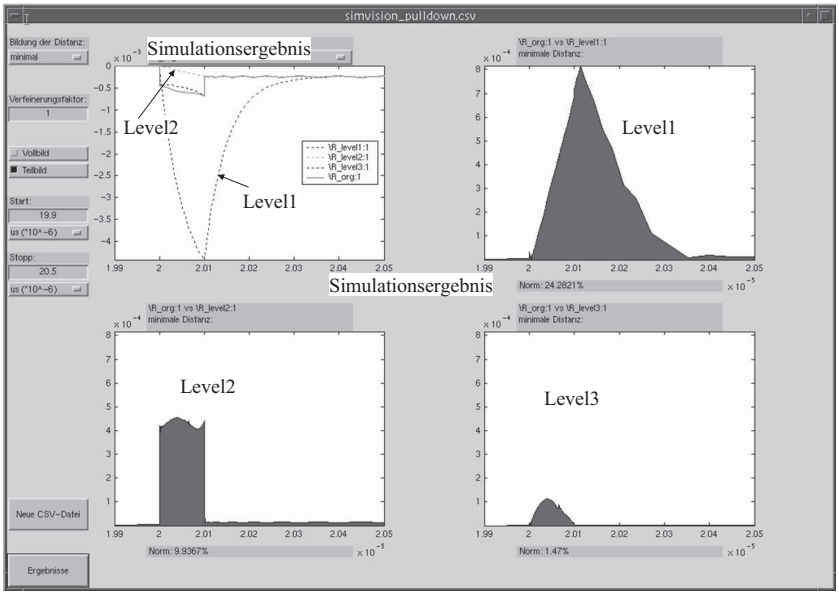

Abbildung 4.13: Modellabweichung beim Ausgangsstrom der Treiberstufe aus Bild 4.6

Kapitel 5

Demonstrator

Im folgenden Kapitel werden die in dieser Arbeit vorgestellten Methoden, anhand eines realen Beispiels für Schaltungs- und Testentwicklung, angewendet. Bei dem Demonstrator, schematisch dargestellt in Bild 5.1, handelt es sich um einen Antennentreiber für niedrige Frequenzen. Er steuert Antennen an, die in automatischen Systemen für die Fahrzeugschlossentriegelung (Passive Entry/Go Systeme) [Schm98] verwendet werden. Der IC beinhaltet Funktionseinheiten wie High-Side/Low-Side Treiber, einen Sinusgenerator, DC-DC Aufwärtswandler (=Boost-Converter), eine Messeinheit und einen Oszillator. Bei der Erstellung des Verhaltensmodells

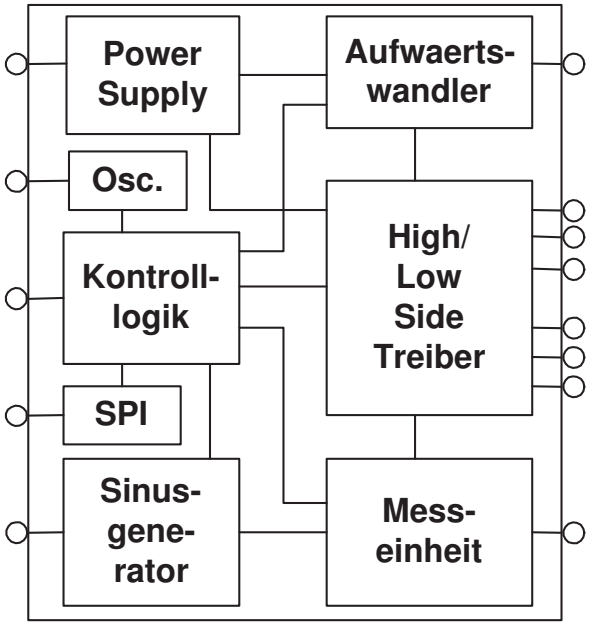

Abbildung 5.1: Blockschaltbild Demonstrator

sind Bibliothekselemente, wie in Kapitel 4.4 beschrieben, verwendet worden. Hierbei handelt es sich um Eingangsstufen, Ausgangsstufen, Pad-Schutzstrukturen und bidirektionale Stufen (siehe Bild 5.2), die im Laufe dieser Arbeit entstanden sind. Die Ausgänge der Treiberstufen sowie des Aufwärtswandlers besitzen große Ausgangstransistoren, die sich selbst gegenüber ESD-Störungen schützen und deswegen keine zusätzlichen Schutzstrukturen benötigen.

Die Schaltung wurde mit der „Meet-In-The-Middle“ Entwurfsstrategie entwickelt. Begleitend

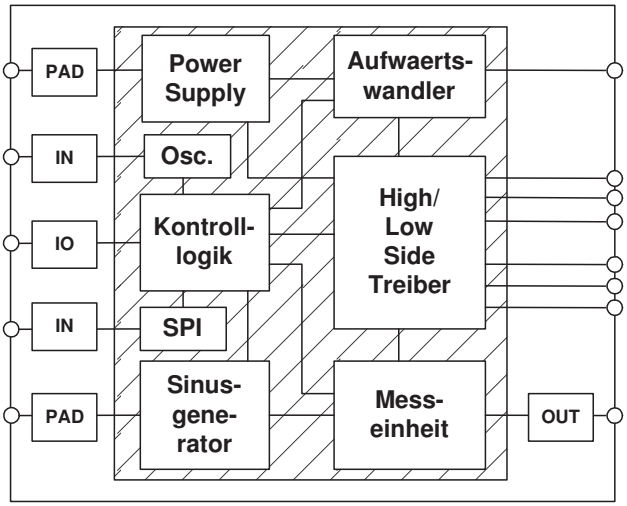

Abbildung 5.2: Blockschaltbild Demonstrator mit Pad und ESD-Schutzstrukturen

zur Schaltungsentwicklung sind Verhaltensmodelle auf den verschiedenen Modell-Level entstanden.

5.1 DC-DC Aufwärtswandler

Im nachfolgenden Abschnitt soll stellvertretend für alle Komponenten des Schaltkreises, der Aufwärtswandler näher vorgestellt werden. Der Wandler wird zum einen für die Anwendungen des virtuellen Tests und zum anderen für die Anforderungen bei der Schaltungsentwicklung aufbereitet. Die Aufgabe des Aufwärtswandlers ist es, von einer Eingangsspannung (ca. 12V) eine größere Ausgangsspannung zu erzeugen (ca. 36-38V). Bei dem Wandler handelt es sich um ein Schaltregler, der durch ein- und ausschalten der Ausgangstransistoren betrieben wird [Tie99].

5.1.1 Schaltungsentwicklung

Der Aufwärtswandler wurde mit einer Meet-In-The-Middle Entwurfsstrategie entwickelt. Am Anfang sind die wichtigsten Kenngrößen der Schaltung, wie beispielsweise die Endspannung,

der Strom welcher am Ausgang benötigt wird, Verlustleistung (=$R_{DS,on}$) und die Güte die das Ausgangssignal beinhalten darf, zu definieren. Hieraus entsteht ein vereinfachtes Level1-Modell, bei dem der Ausgangstransistor mit einem schaltbaren Widerstand nachgebildet wird. In Bild 5.3 ist das Level1-Modell, mit der Ansteuerlogik, einer Diode und einem Schalter dargestellt. Die Diode soll die Bulk-Diode repräsentieren, welche eine Anforderung und somit eine Modelleigenschaft aus der Testentwicklung ist. Auf diese Simulationsaufgabe wird im nächsten Abschnitt (Anwendung für den VT) näher eingegangen.

In dem Modell der Ansteuerlogik befindet sich unter anderem die Funktion der Spannungsbegrenzung. Das Ausgangssignal dieser Logik, welches den Schalter ansteuert, ist digital realisiert. Die Modelleigenschaften für das Level1-Modell und somit die Menge L_1 sind folgende:

- Ausgangsstrom
- Ausgangsspannung (Endspannung)
- Verlustleistung ($R_{on,Schalter}$)
- Taktfrequenz für das Ein- und Ausschalten
- Bulk-Dioden Spannung (ME aus Testentwicklung)

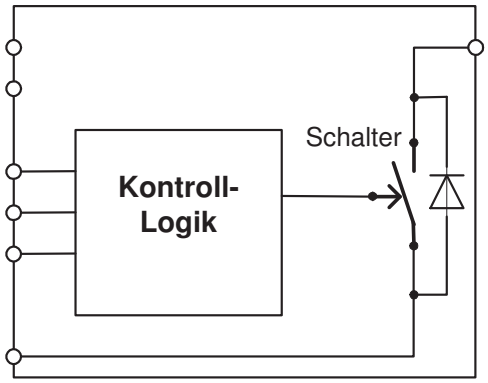

Abbildung 5.3: Blockschaltbild Aufwärtswandler Level1-Modell

Im nächsten Schritt werden die Kenngrößen technologiespezifisch (hier ist es eine SOI-Technologie [Udr00]) zur Weiten- und Längenberechnung der Ausgangstransistoren verwendet. Daraus entstand das Level2-Modell des Aufwärtswandlers (siehe Bild 5.4). Im Gegensatz zum Level1-Modell wird der Schalter durch einen realen Transistor ersetzt. Zusätzlich beinhaltet das Level2-Modell noch eine Überstromerkennung mit der Aufgabe bei Überstrom (hier 1 Ampere) an dem

Ausgangstransistor die Stufe abzuschalten. Die Ansteuerlogik muss ebenfalls so verändert werden, damit das Ausgangssignal, welches zur Gate-Ansteuerung des Ausgangstransistors verwendet wird, einen analogen Spannungsverlauf zugewiesen bekommt. Weiter wird noch ein Strombegrenzer vor dem Gate des Ausgangstransistors benötigt, der das Umladen der Gatekapazitäten näher an der Realität nachbilden soll. Das resultierende Blockdiagramm ist in Bild 5.4 dargestellt.

Die Modelleigenschaften für die Menge L_2 und somit für das Level2-Modell sind nachfolgend

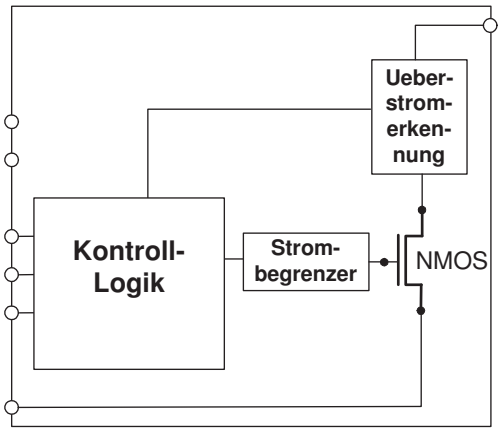

Abbildung 5.4: Blockschaltbild Aufwärtswandler Level2-Modell

aufgelistet.

- Modelleigenschaften aus $\widehat{L_1}$
- Strombegrenzung am Ausgang
- Gatespannung für die Ausgangstransistoren
- Gatestrom-Nachbildung
- Biasstrom zum Aktivieren der Komponente
- Bandgapspannung als Referenzspannung
- Digitale Steuersignale

Bild 5.5 zeigt das Level3-Modell, bei dem ein kompletter Regelkreis implementiert ist. Dieses Modell beinhaltet das AC-Verhalten der Originalschaltung und ist optimal für Stabilitätsuntersuchungen im Designprozess geeignet. Auch die Peripherie ist jetzt identisch zur Transistorschaltung. In dieser fortgeschrittenen Designphase sind jetzt für alle Subblöcke wie beispielsweise

den Fehlerverstärker (Error-Amplifier) die Modelleigenschaften bekannt. Dies war in der frühen Phase der IC-Entwicklung noch nicht möglich.

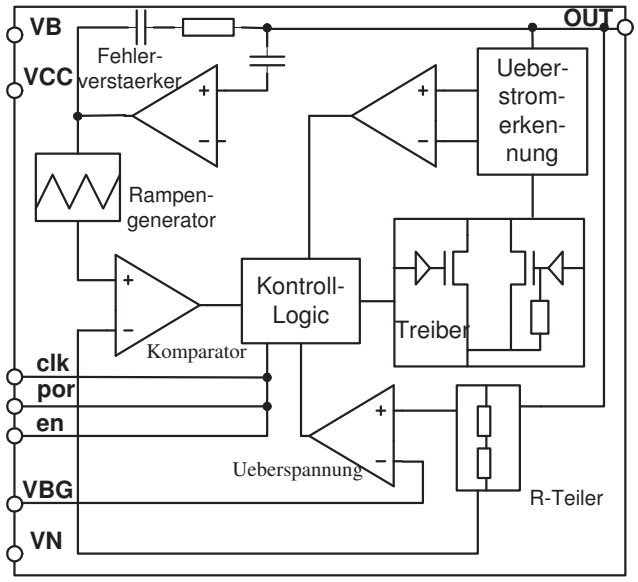

Abbildung 5.5: Blockschaltbild Aufwärtswandler Level3-Modell

Der Ausgangstreiber, siehe Bild 5.6, ist im Level3-Modell als Transistorschaltung realisiert. Die Modelleigenschaften für das Level3-Modell sind folgende:

- Modelleigenschaften aus $\widehat{L_1}$ und $\widehat{L_2}$
- AC-Verhalten (Nachbilden des Regelkreises)
- Güte der Ausgangsspannung bzw. Strom

Diese globalen Eigenschaften werden jetzt auf die einzelnen Subblöcke des Aufwärtswandlers übertragen. Zur Verdeutlichung soll an dieser Stelle die Eigenschaften des Fehlerverstärkers dargestellt werden. Folgende Eigenschaften sind in $L_{3,Fehlerverstaerker}$ enthalten:

- Eingangswiderstand
- Ausgangswiderstand
- Verstärkung
- AC-Übertragungsfunktion (Pool- und Nullstellen)

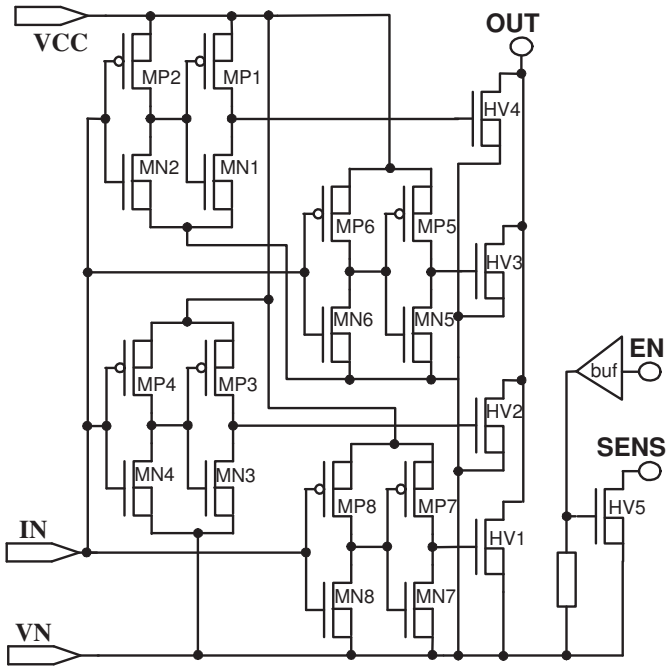

Abbildung 5.6: Transistorschaltbild Treiberstufe

- Ausgangsspannung begrenzend auf Betriebsspannung

In der Phase, als die Schaltung auf Transistorebene realisiert wurde, wurden die Pool- und Nullstellen durch eine Pool- und Nullstellenanalyse ermittelt und mit einer Bottom-Up-Strategie in das Modell implementiert. Diese Übertragungsfunktion lautet wie folgt:

$$F(s) = g0 * \frac{1.0}{1.0 + (1/(2 * Pi * f0))s} \tag{5.1}$$

Es handelt sich hierbei um ein Tiefpassverhalten 1. Ordnung.
Im nachfolgenden Source-Code ist der Fehlerverstärker dargestellt.

```
‘include "constants.vams"
‘include "disciplines.vams"

module error_opamp(OUT, INN, INP, VSS, VDD);

  inout OUT, INN, INP, VSS, VDD;
  electrical OUT, INN, INP;
  electrical VSS, VDD;
  parameter real gain = 2.37k   from (0:inf);
  parameter real F0   = 144K    from (0:inf);
```

```
  parameter real Rin  = 10M    from [0:inf);
  real vout;
analog   begin
  //--------   Eingangswiderstand  ----------------
    V(INP, INN) <+ I(INP, INN) * Rin;
  //--------    Uebertragungsfunktion   -----------
  vout = laplace_nd(V(INP, INN)* gain,{1.0},{1.0, 1/(2*'M_PI*F0)});
  //-------------  SIGNAL CLAMPING   -------------
  vout =  max(V(VSS), vout);
  vout =  min(V(VDD), vout);
  V(OUT) <+  vout;
end
endmodule
```

Das AC-Simulationsergebnis ist in Bild 5.7 zu sehen. Die 3dB-Grenzfrequenz liegt bei ca. 144KHz und die Verstärkung bei ca. 2,37K was 67,5dB entspricht. Dies stellt natürlich nur eine erste Näherung gegenüber der originalen Übertragungsfunktion dar, was aber für die Anforderungen an das Verhaltensmodell völlig ausreichend ist. Zudem ergeben sich erst bei höheren Frequenzen größere Abweichungen, welche aber in dieser Applikation nicht von Bedeutung sind.

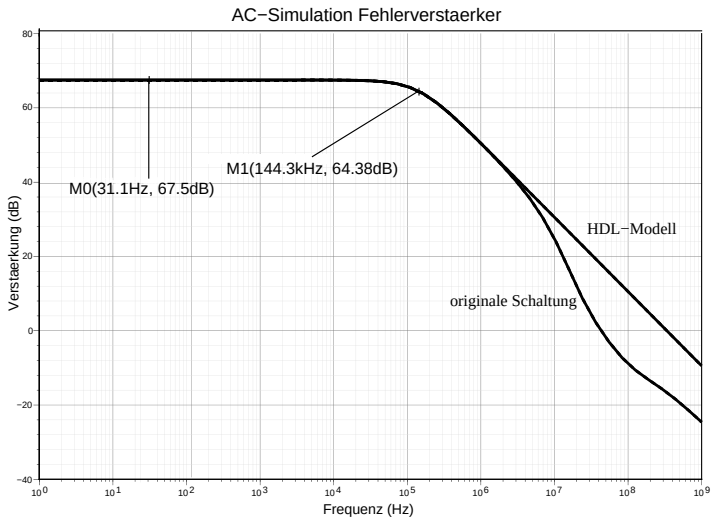

Abbildung 5.7: AC-Simulationsergebnis des Fehlerverstärkers (Vergleich Original zum Modell)

Alle anderen Komponenten des Aufwärtswandlers wurden auf die selbe Art und Weise wie der Fehlerverstärker realisiert. In Bild5.8 ist das Simulationsergebnis des gesamten Wandlers mit den verschiedenen Level-Modellen und der originalen Transistorschaltung dargestellt. Die dazu

gehörenden Simulationszeiten und die Modellgenauigkeiten befinden sich in Tabelle 5.1.

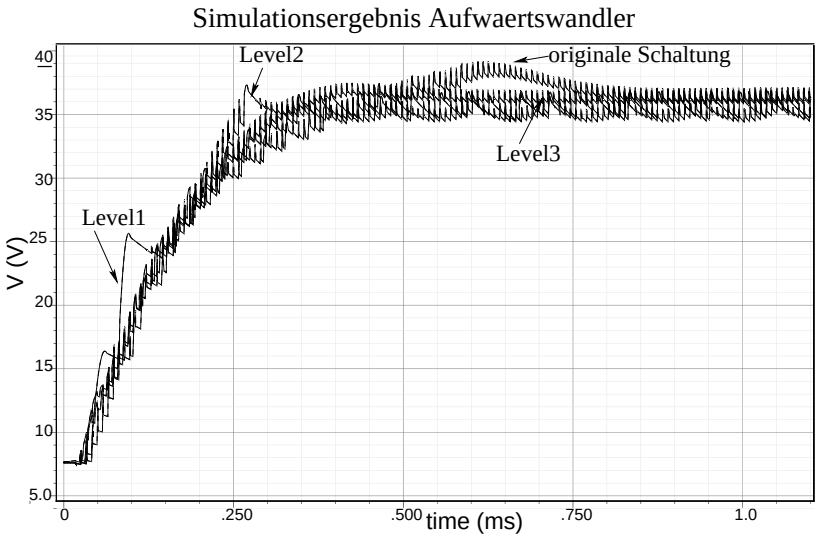

Abbildung 5.8: Simulationsergebnis Aufwärtswandler

Im Ergebnis ist das Hochlaufverhalten des Aufwärtswandlers vom Level1, Level2 und Level3-Modell ersichtlich. Als Referenzkurve ist das Simulationsergebnis der originalen Schaltung mit dargestellt. Da es sich hierbei um einen Schaltregler handelt, ist das Ein- und Ausschalten der Treiberstufe im Kurvenverlauf, selbst im eingeschwungenen Zustand, ersichtlich. Die Modellgenauigkeit wurde, wie in Kapitel 4.5 beschrieben, durch das Tool SEAP ermittelt. Als Simulator ist der AMS-Designer von Cadence verwendet worden.

	Level3	**Level2**	**Level1**
Performancegewinn	92 x	181 x	201 x
Modellabweichung	7,2%	13,4%	17,8%

Tabelle 5.1: Simulationsperformance und Modellgenauigkeit des Boost-Converters

Ist die Phase der Schaltungsverifikation erreicht, stehen alle Modelleigenschaften sowie die Topologie der originalen Transistorschaltung und deren Realisierung zur Verfügung. Wie zuvor beschrieben sind die Verhaltensmodelle des Aufwärtswandlers jetzt als Level1 bis Level3-Modelle realisiert. An dieser Stelle werden nun einige Subblöcke des Level3-Modells durch konfigurierbare Modelle ausgetauscht, siehe hierzu Kapitel 3.3.5. Um den Modellierungsaufwand so gering wie möglich zu halten werden in der Praxis nicht alle Subblöcke durch konfigurierbare HDL-Modelle ersetzt sondern nur die, die in ihrer vollen Funktionalität rechenintensiv

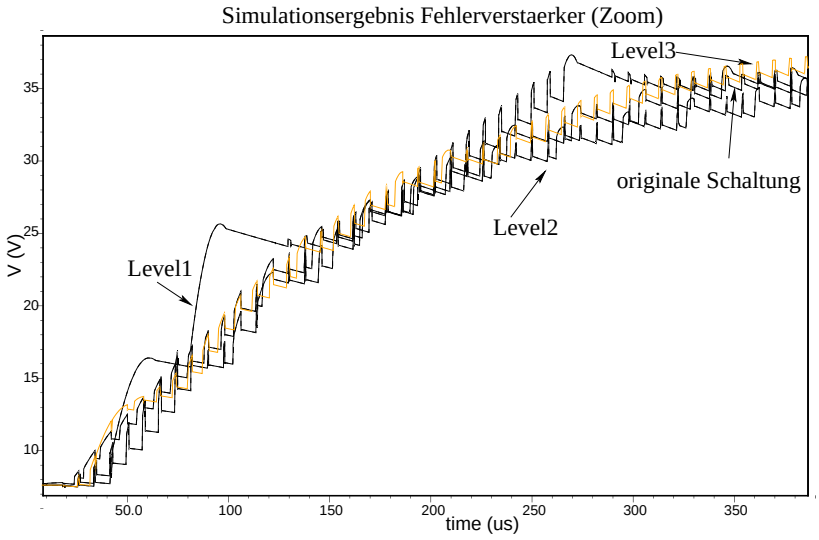

Abbildung 5.9: Simulationsergebnis Aufwärtswandler (Zoom)

sind. Hierzu zählt beispielsweise auch der Fehlerverstärker, der im nachfolgenden Beispiel als konfigurierbares Modell überarbeitet werden soll.

Zunächst werden alle Eigenschaften $E = \{e|e_1, ..., e_i\}$ des Verstärkers gesammelt und klassifiziert. Diese Eigenschaften sind im Modell ein- bzw. ausschaltbar realisiert worden. Im nachfolgenden Source-Code ist dies für den Fehlerverstärker umgesetzt worden.

```
‘include "constants.vams"
‘include "disciplines.vams"

module error_opamp(OUT, INN, INP, VSS, VDD);
  inout OUT, INN, INP, VSS, VDD;
  electrical OUT, INN, INP;
  electrical VSS, VDD;
  parameter real gain = 2.37k   from (0:inf);
  parameter real F0   = 144k    from (0:inf);
  parameter real Rin  = 10M    from [0:inf);
  real vout;
analog   begin
´ifdef Eingangswiderstand
    V(INP, INN) <+ I(INP, INN) * Rin;
´endif
´ifdef Uebertragungsfunktion
    vout = laplace_nd(V(INP, INN)* gain,{1.0},{1.0, 1/(2*‘M_PI*F0)});
´else
```

```
  vout = V(INP, INN)* gain;
´endif
´ifdef CLAMPING
    vout =  max(V(VSS), vout);
    vout =  min(V(VDD), vout);
´endif
    V(OUT) <+  vout;
end
endmodule
```

Da sich nicht mehrere Instanzen des Fehlerverstärkers in dem Schaltkreis befinden, ist hier die Präprozessoranweisung - Realisierung mit *´ifdef* gewählt worden.

Weiter wurden noch die Subblöcke vom Komparator, vom OpAmp der Überstromerkennung sowie der OpAmp der Überspannungserkennung durch konfigurierbare Modelle ausgetauscht. In Tabelle 5.2 sind die Simulationsergebnisse des konfigurierbaren Level3-Modells des gesamten Aufwärtswandlers, einmal mit und einmal ohne AC-Verhalten (=Laplace Übertragungsfunktion), dargestellt.

Modelleigenschaft	**Performancegewinn**
mit Übertragungsfunktion	92 x
ohne Übertragungsfunktion	152 x

Tabelle 5.2: Performancegewinn des Aufwärtswandlers mit und ohne Übertragungsfunktion

Nachfolgend wurden die Transistoren der Treiberstufen durch konfigurierbare MOSFET HDL-Modelle im Level2 und Level3-Modell ausgetauscht. Der Performanceunterschied zwischen optimal konfigurierten MOSFET HDL-Modellen und Modellen mit vollem Funktionsumfang ist in der nachfolgenden Tabelle dargestellt.

MOSFET HDL-Modelleigenschaft	**Level1**	**Level2**	**Level3**
voller Funktionsumfang	201 x	181 x	92 x
optimal konfiguriert	786 x	719 x	123 x

Tabelle 5.3: Performancegewinn des Aufwärtswandlers mit und ohne optimal konfigurierten MOSFET HDL-Modellen

Der Modellfehler für die Simulationsaufgabe mit optimal konfigurierten MOSFET HDL-Modelle ist in Tabelle 5.4 dargestellt. Die Ergebnisse sind auch in dem Betrag [Web07B] veröffentlicht.

Modelllevel	Performancegewinn	Modellabweichung
Level1	786 x	19,6%
Level2	719 x	16,1%
Level3	123 x	9,4%

Tabelle 5.4: Performancegewinn und Modellabweichung des Aufwärtswandlers mit optimal konfigurierten MOSFET HDL-Modellen

Die Konfiguration der MOSFET HDL-Modelle, für die Treiberstufe im Level3-Modell (siehe Bild 5.6) ist wie folgt eingestellt worden (siehe Tabelle 5.5):

Variante	Mx1,3,5,7	Mx2,4,6,8	HV1,2,3,4,5
var	1	1	2
res_nodal	nein	nein	ja
cap_gate	nein	nein	ja
cap_sub	nein	nein	ja
dio_sub	nein	nein	ja
threshold	nein	nein	ja

Tabelle 5.5: MOSFET HDL-Modellkonfiguration

5.1.2 Anwendung für den virtuellen Test

Das Modell soll nicht nur für die Schaltungsentwicklung, sondern auch für den virtuellen Test verwendet werden. Hierbei ist nicht das Ziel, das Testprogramm, welches am Tester für den IC zum Einsatz kommt, wie in Kapitel C beschrieben zu testen. Vielmehr soll anhand der Testbench, die auf Basis der Prüfvorschrift entstanden ist, das Testkonzept bzw. die Testmethodik am Rechner simuliert und analysiert werden. Dieser Ansatz ist in [Leh07] genauer beschrieben.

Die Modelleigenschaften, die für den VT implementiert werden müssen, sind auf Basis der Prüfvorschrift zu ermitteln. Da für den Aufwärtswandler nur der Ausgangspin getestet werden kann (alle anderen Pins sind intern), gibt es nur eine begrenzte Anzahl von Eigenschaften, die in der Treiberstufe zu realisieren sind. Durch den dortigen Einsatz von konfigurierbaren MOSFET HDL-Modellen, wie in Kapitel 4.3 erläutert, muss die Konfiguration der Modelle je nach Simulationsaufgabe unterschiedlich angewandt werden. Im nachfolgenden Abschnitt sollen anhand einiger Anforderungen aus der Prüfvorschrift des Schaltkreises, das optimale Verwenden und Einstellen der konfigurierbaren MOSFET HDL-Modelle demonstriert werden. Das Ermitteln des $R_{DS,on}$, der Bulk-Diodenspannung (für Open/Short- Test [Ant02]), den Ausgangsstrom

und die Bestimmung der Endspannung sind die Simulationsaufgaben für die in Tabelle 5.6 dargestellten Ergebnisse. Für die Simulationsaufgaben, Ermittlung der Endspannung und der Bulk-Diodenspannung, wird das Level1-Modell und für die Aufgabe $R_{DS,on}$ und den Ausgangsstrom wird das Level2-Modell des Konverters verwendet. Dieser Entscheidungsprozess muss je nach Simulationsaufgabe und somit nach der Anforderung an das Modell manuell durchgeführt werden.

Simulationsaufgabe	**Modell-Level**	**Performancegewinn**	**Modellabweichung**
Ermittlung von $R_{DS,on}$	Level2	488 x	15%
Ausgangsstrom	Level2	221 x	10%
Endspannung	Level1	604 x	10%

Tabelle 5.6: Simulationsperformance und Modellgenauigkeit des Aufwärtswandlers mit konfigurierbaren MOSFET HDL-Modellen

Die minimale und maximale zulässige Abweichung der einzelnen Ergebnisse ist in der Prüfvorschrift definiert (siehe Tabelle 5.7). Diese Werte stellen die Anforderung an die Modellgenauigkeit und somit an die Auswahl der verschiedenen Modell-Level bzw. Eigenschaften der konfigurierbaren MOSFET HDL-Modelle dar. Ein Test wird als bestanden definiert, wenn das

Simulationsaufgabe	**min**	**typ**	**max**
Ermittlung von $R_{DS,on}(inOhm)$	0.4	0.6	0.8
Ausgangsstrom (in A)	0.8	1.0	1.2
Endspannung (in V)	33	36	39

Tabelle 5.7: Auszug aus der Prüfvorschrift

Ergebnis zwischen dem minimalen und maximalen Wert liegt.

5.2 Gesamtsystem

Abschließend soll der gesamte Schaltkreis im Fokus stehen. Einerseits mit der Verifikation beim IC-Design (=Gesamtsimulation) andererseits durch simulative Überprüfung von Testmethoden für die Testentwicklung.

5.2.1 Schaltungsverifikation des ICs

Ziel der Gesamtsimulation ist es Fehler zu finden, die meist bei der Interaktion mehrerer Blöcke auftreten. Eine Simulation auf Transistorebene ist bei diesem Schaltkreis nur schwer möglich.

Es handelt sich hierbei um einen analogen bzw. Mixed-Signal ICs mit über 30.000 Instanzen, die auf Transistorebene schon für das Einstellen der internen Spannungen und der Biasströme mehrere Tage Simulationszeit benötigt. Weiter dauert die Simulation der Ausgangstreiberstufen ca. 3 Tage.

Der Einsatz von Verhaltensmodellen ist hierbei unumgänglich. Mit der selben Methodik, wie der zuvor beschriebene Aufwärtswandler, wurden für alle anderen Blöcke des Schaltkreises Verhaltensmodelle mit verschiedenen Modell-Level realisiert. Die Simulationszeiten der einzelnen Blöcke sind in Tabelle 5.8 zu sehen. Bei den einzelnen Blöcken konnte ein Performancegewinn bis zu 1.351 x erzielt werden.

Durch die Realisierung der verschiedenen Modell-Level sowie der Einsatz von konfigurierba-

Komponente	**Level1**	**Level2**	**Level3**
Sinusgenerator	1296 x	311 x	152 x
Messblock	619 x	316 x	215 x
Supplyblock	598 x	366 x	310 x
Ausgangstreiber	1351 x	653 x	349 x

Tabelle 5.8: Simulationsperformancegewinn der Schaltkreiskomponenten

ren Modellen können bei der Gesamtverifikation je nach Simulationsaufgabe gezielt Modelleigenschaften aktiviert oder deaktiviert werden. Mit dieser Voraussetzung war eine erfolgreiche Verifikation des ICs möglich.

Nun soll stellvertretend für alle Verifikationsaufgaben die Funktion des Schaltkreises analysiert werden. In Bild 5.10 ist eine Gesamtsimulation mit Level1-Modellen zu sehen. Bei dieser Simulationsaufgabe wird die Antenne-Nr.2 aktiviert. Weiter ist das Hochlaufen der Spannung VDS, welches die Ausgangsspannung des Aufwärtswandlers repräsentiert, dargestellt. Die internen Versorgungsspannungen CINT und VCC, die Sinusgeneratorspannung VShunt sowie digitale Signale sind ebenfalls im Simulationsergebnis enthalten. Die digitale Signale sind MACT = aktivieren der Ausgangsstufe; BCNT = digitales Eingangssignal für die Ausgangsstufe; IRQ = Kommando das über eine SPI übertragen wurde ist als gültiger Befehl vom Digitalteil erkannt worden.

Diese Simulation soll nur das prinzipielle Zusammenspielen der Blöcke veranschaulichen. Mit einer Simulationszeit von ca. 16 Minuten stellt dies die Konfiguration mit dem größten Performancegewinn dar. Ein Vergleich mit der originalen Transistorschaltung ist hier nicht möglich, da die Simulationsdauer mehrere Wochen oder Monate dauern würde. Verwendet wurde ein 2,8GHz Dual-Core Rechner mit 4 GB RAM. Die Verifikationsstrategien sind nicht Inhalt der Arbeit. Diese wird zur Zeit im Förderprojekt „VeronA“ erarbeitet und evaluiert.

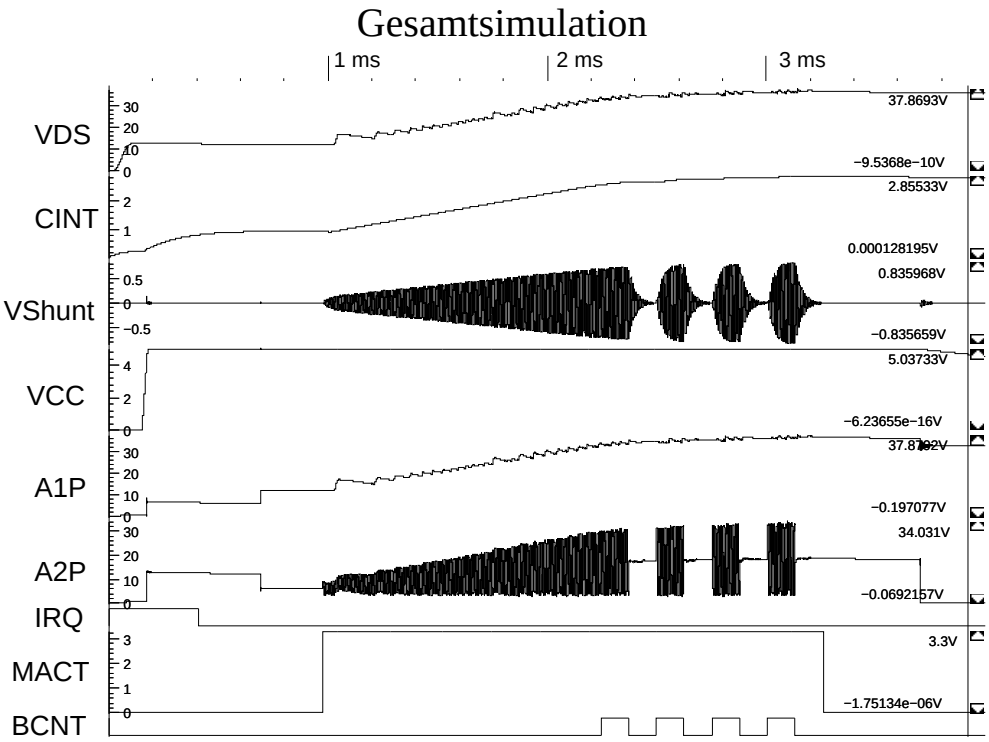

Abbildung 5.10: Gesamtsimulation Demonstrator

Konfigurationsbeispiele einiger Verifikationsaufgaben sind in der Tabelle 5.9 zu sehen. Bei diesen Simulationsergebnissen sind die konfigurierbaren HDL-Modelle für die jeweilige Aufgabe „optimal“ eingestellt worden.

Durch das Vorhandensein der Level-Modelle und deren Konfigurationsmöglichkeiten, konnte mit optimierter Simulationszeit gezielt die Verifikation der Schaltung vorgenommen werden. Gegenüber dem „Stand der Technik“-Modellierungsansätzen wird bei jeder Verifikationsaufgabe Simulationszeit, bei der vorhandenen Genauigkeit, eingespart. Aufgrund von „time-to-market“ Zielen oder Vorgaben in der Industrie, steht nach der Fertigung der Schaltung nur noch eine begrenzte Zeit für die Gesamtverifikation zur Verfügung. Diese Zeit kann jetzt zum Verifizieren und somit zur Steigerung der Designsicherheit genutzt werden.
Die Fehler, die durch die Gesamtsimulation entdeckt und behoben wurden, führten zu einem funktionsfähigen Silizium in der Version 1.0.

5.2.2 Simulation der Prüfvorschrift

Für den virtuellen Test wird zusätzlich zu dem Verhaltensmodell ein Loadboard Modell, das vereinfacht das Verhalten des realen Loadboard wiedergibt, benötigt. Ein Signalgenerator, der au-

Sim.-aufgabe	System-funktion	Aufwärts-wandler	Mess-block	Sinus-generator	Ausgangs-treiber	Oszil-lator
Sinus-generator	Level1	Level1	Level2	orig.	Level3	Level1
Messblock	Level1	Level1	orig.	Level2	Level1	Level1
Supply-block	Level1	Level3	Level3	Level3	Level3	Level2
Ausgangs-treiber	Level1	Level3	Level2	Level1	orig.	Level1
Oszil-lator	Level1	Level1	Level2	Level3	Level1	orig.
Aufwärts-wandler	Level1	orig.	Level3	Level1	Level3	Level1
Kontroll-Logik	digital	digital	digital	digital	digital	digital
Sim.-zeit	16min	2std	4,5std	7,5std	67std	23std

Tabelle 5.9: Konfigurationsbeispiele für die Gesamtverifikation des ICs

tomatisch aus der Prüfvorschrift Verilog-AMS Stimuli generiert sowie ein Auswertungsmodell, werden ebenso benötigt. Diese Modelle sind in Laufe der hier beschriebenen Arbeit entstanden. Eine genauere Beschreibung befindet sich in [Leh07]. In Bild 5.11 ist das Blockschaltbild mit Loadboard, Signalgenertator und Auswerteeinheit dargestellt.

Es wurden die Testreihen der Funktionseinheit Aufwärtswandler mit dem AMS-Designer der Firma CADENCE simuliert. Um die erforderlichen Signalquellen für die einzelnen Tests mit dem Signalgenerator zu erzeugen, wurden die entsprechenden Daten aus der Prüfvorschrift in die Textdatei extrahiert. Ebenso wurde mit den Messdaten (Einheiten, Grenzwerte) welche die Messwerterfassung benötigt, verfahren. Die Ergebnisse der Tests, Messwert, Einheit, obere Grenze, untere Grenze sowie das Urteil „passed“ oder „failed“ wurden wiederum in eine weitere Textdatei geschrieben (siehe hierzu [Leh07]).

Die Konfiguration der Modelle wurde je nach Simulationsaufgabe unterschiedlich vorgenommen. Als Beispiel wurde hier ebenfalls wie in Kapitel 5.1.2 $R_{DS,on}$, die Endspannung und der Ausgangsstrom des Aufwärtswandlers analysiert. DerUnterschied ist, das jetzt der gesamte Schaltkreis mit Loadboard, Signalgenerator und Auswertungsmodell involviert ist. Bei der Konfiguration werden alle anderen Blöcke durch Level1-Modelle ersetzt. Der Aufwärtswandler

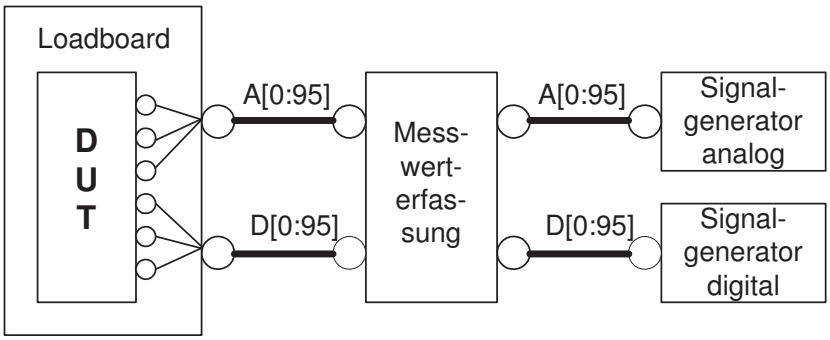

Abbildung 5.11: Testbench für VT-Simulationen

selbst, wird wie in Tabelle 5.10 konfiguriert. Die konfigurierbaren HDL-Modelle wurden wie in dem Kapitel 5.1.2 eingestellt.

Simulationsaufgabe	**Modell-Level**	**Simulationszeit**
$R_{DS,on}$	Level2	9min
Ausgangsstrom	Level2	9min
Endspannung	Level1	2min

Tabelle 5.10: Simulationszeiten und Modellgenauigkeit des Gesamtsystems

Mit der hier beschriebenen Methode wurde in einer frühen Phase simulativ einige kritische Tests aus der Prüfvorschrift analysiert. Durch die „entwurfsablaufübergreifende Modellierungsmethodik“ mussten speziell für die Testsimulationen keine Modellanpassungen oder sogar eine neue Modellierung des DUT vorgenommen werden.

5.3 Zusammenfassung

Für den hier vorgestellte Demonstrator (Antennentreiber-IC für Passiv Entry/Go Systeme) wurden Verhaltensmodelle erstellt und verwendet, die sowohl für die Design- und Verifikationsphase bei der IC-Entwicklung, als auch für die Simulationen in der Testentwicklung benötigt werden.

Bei der Schaltungsentwicklung, wurde begleitend zur einer Meet-In-The-Middle Entwurfsstrategie die Level1 bis Level3 Verhaltensmodelle entwickelt. Nach Anwendung einer Bottom-Up-Strategie, wurden Subblöcke des Level3 und Level2-Modells durch konfigurierbare HDL-Modelle ausgetauscht.

Die Modellerstellung wurde stellvertretend für alle Blöcke des ICs an dem DC-DC Aufwärtswandler veranschaulicht. An dem Wandler konnte gezeigt werden, dass je nach Simulationsaufgabe durch die Kombination von den Modell-Leveln und den konfigurierbaren Modellen eine

gezielte Modellgenauigkeit und einen verbesserten Performancegewinn erreicht werden kann. Hierzu hat u.a. der Einsatz von konfigurierbaren MOSFET HDL-Modellen beigetragen.

Weiter wurde das Modell des Aufwärtswandler so entwickelt, dass dieses auch die Anforderungen bei der Testentwicklung mit einbezieht. Eine optimale Konfiguration des HDL-Modells wurde hier am Beispiel drei verschiedener Tests gezeigt. Das Ziel war einen Performancegewinn, mit dem selben Modell welches bei der Schaltungsentwicklung verwendet wurde, zu erreichen.

Am Ende des Kapitels wurden Ergebnisse des gesamten Schaltkreises für die Schaltungsverifikation und Simulationen einzelner Tests aus der Prüfvorschrift gezeigt. Der Einsatz von HDL-Modellen war bei diesem Schaltkreis notwendig, da die Simulation auf Transistorebene nicht möglich gewesen wäre und einige Wochen bzw. Monate Simulationszeit benötigt hätte. Durch den Einsatz von Verhaltensmodellen wurde eine Gesamtsimulation mit Level1-Modellen in 16 Minuten durchgeführt. Bei der Verifikation konnte durch die konfigurierbaren HDL-Modelle gezielt Eigenschaften aktiviert oder deaktiviert werden. Die Fehler, die durch die Gesamtsimulation entdeckt und behoben wurden, führten zu einem funktionsfähigen Silizium in der Version 1.0. Somit konnten die Kunden frühzeitig mit Mustern ausgestattet werden.

Kapitel 6

Zusammenfassung und Ausblick

Die vorliegende Arbeit stellt eine Modellierungsmethodik die übergreifend im gesamten Entwurfsablauf eingesetzt werden kann vor. Der Schwerpunkt dieser Arbeit liegt bei niederfrequenten Automotive Mixed-Signal ICs, die bevorzugt mit einer Meet-In-The-Middle Entwurfsstrategie entwickelt werden können. Digitale Schaltkreise mit hoher Schaltungs- und Systemkomplexität oder HF-ICs sind bei der Methode nicht berücksichtigt worden.

Der Vorteil der hier vorgestellten Methode gegenüber dem aktuellen Stand der Technik ist, dass bei der Modellerstellung alle im Entwurfsablauf vorkommenden Modelleigenschaften berücksichtigt und implementiert werden, sei es für Simulationen die in der Testentwicklung ablaufen als auch für die Entwicklung und Verifikation der Schaltung.

Die Modellierungsmethodik beinhaltet die Einführung verschiedener Abstraktionsklassen, die sich in ihrer Topologie sowie im Detaillierungsgrad unterscheiden. Diese Klassen werden als „Level1-Modell“ für die höchste, bis „Level3-Modell“ für die niedrigste betrachtete Abstraktionsebene bezeichnet. Begleitend zum Entwurfsablauf werden die genannten Modelle für den Test und für das IC-Design erstellt.

Weiter umfasst die Methodik den Einsatz von konfigurierbaren Modellen, die eher für eine Bottom-Up-Strategie ausgelegt sind. Hier müssen alle Modelleigenschaften und die Schaltungstopologie vor der Modellerstellung bekannt sein, was zum Entwicklungsbeginn (z.B. im Top-Down-Design) nicht der Fall ist. Der Vorteil der konfigurierbaren Modelle ist das Aktivieren und Deaktivieren von Modelleigenschaften. Dies hat zum Ziel, nur die Eigenschaften auszuwählen die für die Simulationsaufgabe bzw. -anforderung notwendig sind und somit den Effekt einen hohen Performancegewinn zu erhalten. Für Simulationen im virtuellen Test ist dies z.B. dringend notwendig.

Durch den Einsatz von Mixed-Level Modellierungsmethoden, die das Verwenden von MOSFET Transistormodellen beinhaltet, wurden konfigurierbare MOSFET HDL-Modelle entwickelt

und vorgestellt. Diese Methode zielt darauf ab, Teile des Transistors zu deaktivieren und somit die Simulation zu beschleunigen.

Die Kombination beider Methoden stellt einen weiteren Beitrag der Arbeit dar. Dies wurde am Beispiel eines Automotive ICs (Antennentreiber für Passiv Entry/Go Systeme) sowohl im Bereich der Schaltungsentwicklung als auch für die Testentwicklung demonstriert.

Die Modellierungsmethodik wurde begleitend zu einer Meet-In-The-Middle Entwurfsstrategie angewandt. Hierbei wurden die Level1 bis Level3-Modelle entsprechend dem Stand der Entwurfsphasen generiert. Nach Fertigstellung des Designs wurde beim Level2 und Level3-Modell einzelne Subblöcke durch konfigurierbare Modelle, wie beispielsweise die MOSFETs, ausgetauscht. Mit diesen konfigurierbaren Level-Modellen konnte eine vielfach verbesserte Performance für Test- und IC-Entwicklung aufgezeigt werden.

Durch diese Art der Modellierung stehen nun hauptsächlich bei der Schaltungsverifikation neue Möglichkeiten zur Verfügung. Nun können Eigenschaften der Schaltung gezielt untersucht werden. Bei einer Gesamtsimulation können beispielsweise einzelne Blöcke auf Transistorebene simuliert werden. Für die interagierenden Blöcke werden konfigurierbare Level3-Modelle verwendet, die für eine realistische Umgebung des zu untersuchenden Blockes beitragen. Alle anderen Blöcke können durch vereinfachte Modelle ersetzt werden.

Ein weiterer Vorteil der konfigurierbaren Modelle ist die Möglichkeit einen Teil im Modell mit hohem und einen anderen Teil mit niedrigem Detaillierungsgrad zu beschreiben, was die Flexibilität des Modells verdeutlicht.

In dieser Arbeit wird das Konfigurieren der Modelle manuell vorgenommen. Aufbauend auf den hier vorgestellten Ergebnissen könnten Methoden, Verfahren oder Regeln für das Konfigurieren erarbeitet werden, die beispielsweise mit Optimierungsalgorithmen für ein Automatismus beitragen.

Das Bewerten eines HDL-Modells ist zur Zeit nur durch manuelles Ermitteln des Performancegewinns und der Modellabweichung möglich. Dem Benutzer sollte eine Möglichkeit gegeben werden, um festzustellen, welche Konfiguration oder welcher Modell-Level der optimale für seine Simulationsaufgabe ist.

Weiter wurde die hier vorgestellte Methode nicht auf Anwendbarkeit für komplexe digitale Systeme oder für HF-Systeme betrachtet.

Ein weiterer offener Punkt stellt die Ermittlung der Modelltopologie für konfigurierbare Level-Modelle dar. Mit den in dieser Arbeit beschriebenen Konfigurationsmethoden können Topologieänderungen im Source-Code vorgenommen werden. Diese Methode scheint aber nur für Subblöcke sinnvoll anwendbar zu sein. Für ein größeres Modell mit verschiedenen Subblöcken (wie das hier beschriebene Level3-Modell des Aufwärtswandlers) wird durch das „Schematic-Entry“ seine Topologie festgelegt. Hier müssen in zukünftigen Arbeiten Regeln abgeleitet und

erstellt werden.

Die Modellerstellung ist wie in dieser Arbeit beschrieben immer noch ein manueller Prozess. Die Umsetzung durch einen Automatismus dürfte schwierig sein. Vielleicht stellt eine Kombination schon bestehender automatischer Modellierungsmethoden mit dieser Methodik ein Verbesserungspotential dar. Zum Beispiel das Ersetzen der konfigurierbaren MOSFET HDL-Modelle durch HDL-Modelle, die auf Basis einer symbolischen Analyse entstanden sind (z.B. mit AI).

Die Idee der konfigurierbaren MOSFET-Modelle könnte auch als Fast-Spice Simulation für analoge bzw. Mixed-Signal Schaltungen verwendet werden. Hierzu müssten die Eigenschaften im MOSFET Simulator Modell (z.B. BSIM oder EKV) aktiviert oder deaktiviert werden können, sodass der Performancegewinn gegenüber dem kompletten MOS-Primitive deutlich sichtbar ist. Dies ist zur Zeit mit der hier beschriebenen Lösung mit Verilog-A trotz C-Compiler die vorher den Code übersetzten, nur schwer möglich.

Ein viel versprechender Ansatz ist, die konfigurierbaren MOSFET HDL-Modelle gleich in einer C-Sprache, die der Simulator optimal interpretieren kann, zu schreiben. Ein Beispiel sind die „CMI“-Modelle für den Spectre Simulator.

Anhang A

Modellklassen

Systeme lassen sich in verschiedenen Zeitmodellen und Abstraktionsebenen beschreiben. Es wird unterschieden zwischen ereignisdiskreter, diskreter und kontinuierlicher Zeit. Graphbasierte, formale Modelle von Systemen erlauben bisher, entweder die Modellierung zeit- und ereignisdiskreter Systemteile oder die Modellierung zeitkontinuierlicher Systemteile. Das Hauptproblem bei der Modellierung von Systemen ist, dass unterschiedliche Zeitmodelle verwendet werden [Gri96]. Für den rechnergestützten Entwurf elektronischer Systeme ist eine formale Spezifikation der geforderten Eigenschaften notwendig. Man unterscheidet zwischen

- textbasierter formaler Spezifikation mit Hilfe von Modellierungssprachen wie z.B. VHDL
- graphischer formaler Spezifikation mit Hilfe von Graphen [Ste93]

Zeitkontinuierliche Teile werden mit Hilfe von Differentialgleichungen oder im Frequenzbereich modelliert (Differential Equation Specified Systems, DESS). Zeitdiskrete Systeme (Discrete Time Systems, DTS) werden durch Automatenmodelle oder durch Differenzgleichungen beschrieben. Ereignisdiskrete Systeme (Discrete Event Systems, DEVS) lassen sich zum Beispiel durch Petri-Netze oder Datenflussgraphen darstellen [Ley95]. Die textbasierte Modellierung, z.B. VHDL-AMS / VHDL-A [IEE93], erlaubt bereits die gemeinsame Modellierung zeitdiskreter oder ereignisgesteuerter Systeme.

Anhang B

Differential-Algebraische Gleichungen

Zeitkontinuierliche Systeme lassen sich mit Hilfe von DAE, z.B. durch eine HDL wie Verilog-AMS, beschreiben. Eine differential algebraische Gleichung besteht aus einer gewöhnlichen Differenzialgleichung gekoppelt mit algebraischen Nebenbedingungen.
Die allgemeinste Form einer differentiell-algebraischen Gleichung ist eine implizite Differenzialgleichung der Form:

$$\dot{\vec{x}} = f(t, \vec{x}, \vec{u}) \tag{B.1}$$

$$g(t, \vec{x}, \vec{y}, \vec{u}) = 0 \tag{B.2}$$

mit:

- $\vec{u}$: Eingangsgröße
- $\vec{y}$: Ausgangsgröße
- $\vec{x}$: Zustandsvariable
- $\dot{\vec{x}}$: $\frac{d\vec{x}}{dt}$

Eine Gleichung in dieser Form ist nur dann nach $\vec{x}$ auflösbar, wenn die partielle Ableitung $f(\vec{x})$ regulär ist. In diesem speziellen Fall kann man die Gleichung in die Form einer expliziten gewöhnlichen Differential-Gleichung umschreiben.

$$\dot{\vec{x}} = A * \vec{x} + B * \vec{u} \tag{B.3}$$

$$\vec{y} = C * \vec{x} + D * \vec{u} \tag{B.4}$$

Anhang C

Virtueller Test

Neben dem Design, dem Layout und der Fertigung spielt, auch der Test integrierter Schaltungen eine wichtige Rolle. Für ein neues Produkt muss ein Testprogramm erstellt werden, das alle Funktionen des Chips prüft. Dies geschieht mit Hilfe von maschinellen Testern, die von einem Programm gesteuert alle Tests am Halbleiter automatisch durchführen. Das heisst, es muss für jedes Produkt ein neues Testprogramm erstellt werden. Dieser Vorgang war seither nur in direkter Verbindung mit vorliegendem Silizium am Tester möglich. Die sequentielle Abfolge aus Schaltungsentwicklung, Fertigung und Testprogrammentwicklung wirkt sich negativ auf die Entwicklungszeit aus. Das folgende Bild C.1 veranschaulicht den zeitlichen Verlauf von der Prüfvorschrift zu einem Testablauf.

Durch die Anwendung des virtuellen Tests kann die Erstellung und Verifikation des Testprogramms größtenteils unabhängig vom realen Halbleiter erfolgen und damit wesentlich früher stattfinden. In Bild C.2 ist diese Parallelisierung der verschiedenen Arbeitsschritte schematisch dargestellt.

Die Grundidee des virtuellen Tests basiert auf der Nachbildung von Testerhardware, DIB und DUT mit Hilfe von Verhaltensmodellen sowie deren Einbindung in eine der Testersoftware entsprechenden Entwicklungsumgebung [Gra99A] [Gra99B]. Die Funktionalität des Testers und der zugehörigen Instrumente werden dabei von den Anbietern der Testerhardware zur Verfügung gestellt, die notwendigen Verhaltensmodelle müssen im Rahmen des Entwurfsprozesses erstellt werden. Im Idealfall werden derartige Modelle im Rahmen einer Top-Down-Designmethodik generiert und können an die Anforderungen im virtuellen Test adaptiert werden.

Wird im Zuge des Entwurfsprozesses kein Verhaltensmodell generiert oder sind der im Design verwendete und der in der Testerumgebung unterstützte Simulator nicht kompatibel, bedeutet die Verhaltensmodellierung einen nicht zu vernachlässigenden zusätzlichen Aufwand. Nach Ablauf der Arbeiten in der virtuellen Testumgebung und der Verfügbarkeit des Siliziums folgt

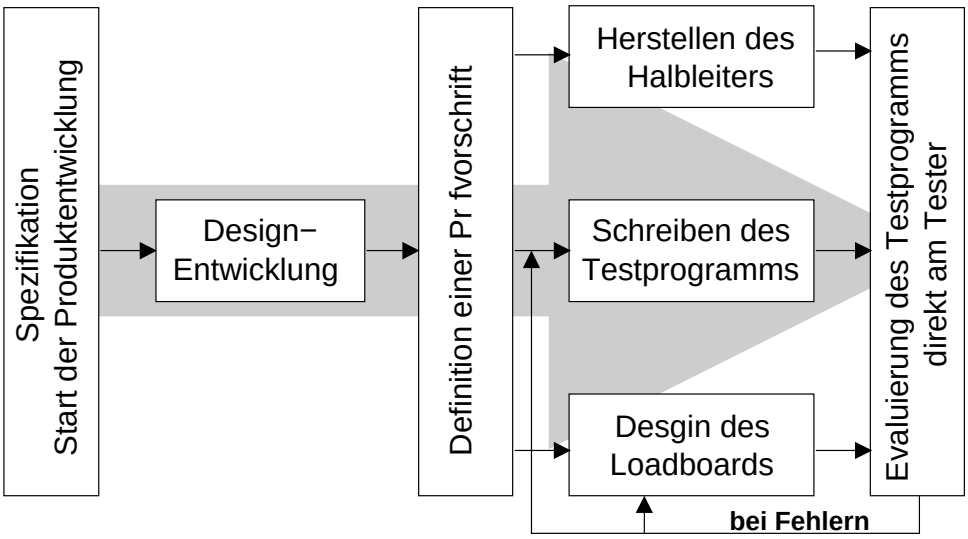

Abbildung C.1: Testprogrammentwicklung

die Inbetriebnahme des Testprogramms am Testautomaten, die sich aufgrund der vorherigen Arbeiten deutlich beschleunigen sollte.

Virtuelle Testumgebung: Die Umgebung VSPACE vom damaligen Anbieter SZ Testsysteme AG, heute Credence, setzt sich im Wesentlichen aus den beiden Komponenten SPACE und einem gekoppelten Simulator zusammen. SPACE (SZ Programming And Control Environment) ist das Softwarepaket, das die Steuerung eines Testers ermöglicht. Dieses Paket umfasst unter anderem die eigentliche Software zur Ablaufsteuerung von Testprogrammen, einen Debugger für Testprogramme, den Kommandointerpreter mit grafischen Panels für die Visualisierung von Testerinstrumenten und verschiedene statistische Tools. Die in VSPACE (Virtual SPACE) integrierte SPACE Version unterscheidet sich in Handling und Funktionalität nicht von der SPACE Version, die zur Offline-Programmentwicklung oder direkt am Tester eingesetzt wird. Lediglich eine zusätzliche Komponente dient der Kommunikation mit dem nachgeschalteten Simulator. In dieser virtuellen Testumgebung wurde der Simulator SABER eingebunden, Verhaltensmodelle mussten entsprechend in der Beschreibungssprache MAST formuliert werden. Im Falle von VSPACE übernimmt der Simulator SABER die Generierung von Stimuli und entsprechender Ausgangssignale auf Basis der zugrundeliegenden Verhaltensmodelle von DUT, DIB und ATE (Automatic Test Equipment). Der Aufruf dieser Stimuli erfolgt durch das Testprogramm, das sich in seiner virtuellen Form nicht von der am realen Testsystem eingesetzten Form unterscheidet. Bild C.3 verdeutlicht das Zusammenspiel der einzelnen Komponenten von VSPACE.

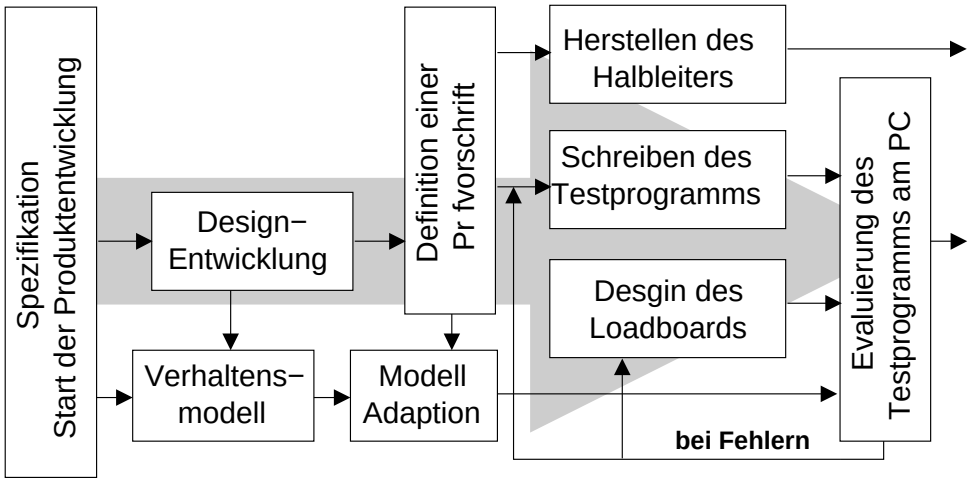

Abbildung C.2: Testprogrammentwicklung mit virtuellen Test

C.1 Stand der Technik von VT Systemen

Der VT findet zunehmend weltweit Anwendung in der Industrie. Die Konzepte werden von verschiedenen Anbietern immer erfolgreicher eingesetzt. Heutzutage wird der VT mit unterschiedlichen Ausführungen angeboten. Hier gibt es z.B. für digitale, analoge oder Mixed-Signal Systeme interessante Realisierungen. Im nachfolgenden Abschnitt werden die Methoden und Konzepte für den virtuellen Test aus weltweiten Quellen vorgestellt und zusammengefasst.

Ein entscheidendes Problem für die Anbieter von VT-Systemen war, dass es beim Entstehen dieser Idee noch keine standardisierte Mixed-Signal Verhaltenssprache wie z.B. VHDL-AMS oder Verilog-AMS gab. Derzeit wurde nur die Sprache MAST diesen Anforderungen gerecht, jedoch mit dem Nachteil, ziemlich stark an den Anbieter (Analogy, Avant!, heute Synopsys) gebunden zu sein. Ein weiterer Nachteil war, dass VHDL, welches schon starken Einzug in der Industrie gefunden hatte, nicht für Mixed-Signal Schaltkreise verwendet werden konnte. Im nachfolgenden Abschnitt werden verschiedene Umgebungen vorgestellt.

Das Förderprojekt VIRTUS [Ein99A] hat sich gezielt mit dem Thema Virtueller Test befasst, sowohl mit der Nachbildung von Testerkomponenten als auch mit der Erstellung von VT-Modellen mit der Hardwarebeschreibungssprache MAST, die für den Simulator SABER implementiert wurden. Desweiteren werden hier Kommunikationskonzepte vorgestellt, die einen optimalen Datenaustausch zwischen Simulator und Softwareumgebung ermöglicht. Weiter wird in dem Projekt eine direkte Anbindung an die Testersoftware realisiert, die dem Testingenieur in der Simulationsumgebung die identische Arbeitsumgebung zur Verfügung stellt wie auf der realen Hardware. Alle Debugging-Optionen können ohne Einschränkung verwendet werden. Die Simulation von Testprogrammen kann wegen der Komplexität eines Testsystems nur auf Verhal-

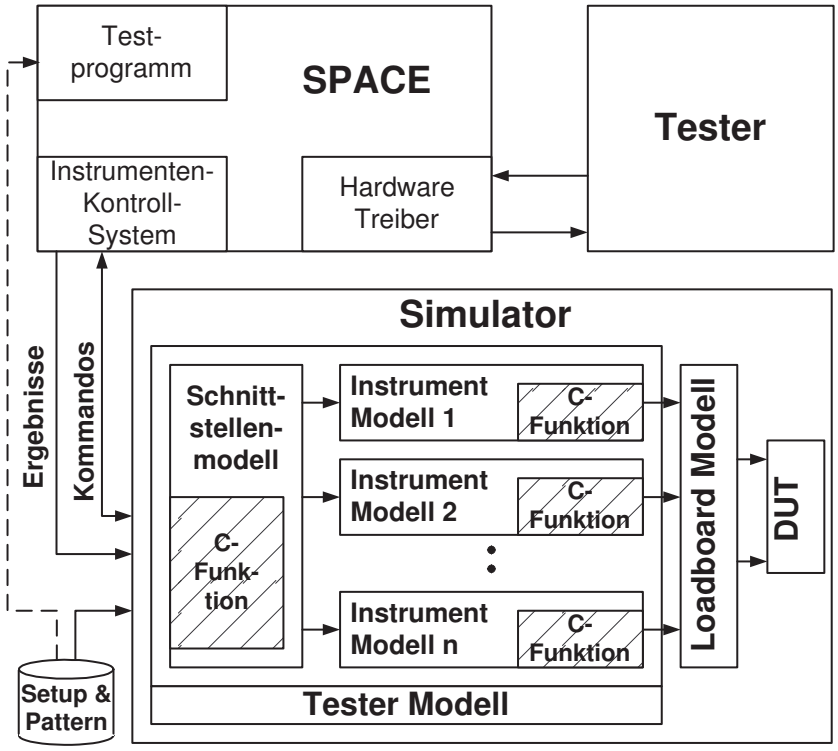

Abbildung C.3: Nachbildung von ATE, DIB und DUT [Gra99A]

tensebene erfolgen [Gra99C]. Während des Testvorgangs fallen große Datenmengen an, welche die Einstellungen für jedes einzelne Instrument, Messwerte, Ergebnisse, Testmuster und Pinlisten sowie Abtastwerte analoger Signale umfassen. Diese Daten werden in geeigneter Weise verwaltet.

„Analog Devices“ haben ihre eigene Testprogrammumgebung entwickelt [Rio99]. Aufbauend auf die Software von Teradyne IMAGE und ADICE. Diese Software unterstützt sowohl die Simulation von DUT (durch ADICE), als auch die Testprogrammentwicklung (durch IMAGE). Gekoppelt werden die zwei Simulatoren durch „Exchange“ und ein Interface Programm. Das Design der Testsimulation wird somit effektiv abgerundet.

Eine digitale Testprogrammentwicklung wird von Dave Rolince von der Firma Teradyne [Rol97] vorgestellt. Hier handelt es sich um einen kompletten Entwicklungszyklus (siehe Bild C.4), angefangen von Verhaltensmodellen in VHDL/Verilog über eine Synthese bis hin zum Testprogramm. Weiter kann man in dem Bild sehen, dass nach der Synthese die Modelle auf Gatter Ebene (Gate level) inklusive VITAL beschrieben werden. VITAL beinhaltet die Verzögerungs- und Durchlaufzeiten der einzelnen Gattern und Leitungen, welche nach der Synthese bestimmt werden können.

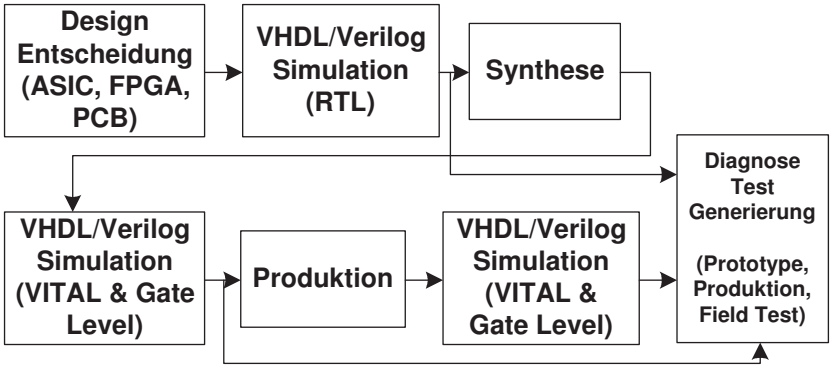

Abbildung C.4: Digitaler Testflow mit Synthese und Simulation mit VITAL [Rol97]

Ein weiterer Ansatz für eine VT-Umgebung stellt die Integration von MATLAB und SIMULINK dar. In [Kle] wird eine Software vorgestellt, die es ermöglicht, virtuell zu testen. Hier werden verschiedene Komponenten auf Softwareebene miteinander verbunden. Die Hauptkontrolleinheit und das User-Interface wurden in C++ programmiert, mit der Aufgabe, MATLAB und SIMULINK für die Modellierung und Simulation zu integrieren. Ein Ziel bei der Entwicklung dieses Software Tools war, Test- und Analyseorganisationen zu unterstützen. Diese haben die Fähigkeit auf mathematischer Ebene die komplette Testumgebung mit dem Computer zu simulieren und ermöglichen somit einen virtuellen Test.
Weiter kann die Modellbibliothek von SIMULINK zur Erstellung des DUT-Modells genutzt werden und erleichtert somit die Modellierung von Standardelementen (z.B. OpAmp's). Nicht nur das mathematische Verhalten sondern auch das zeitliche Verhalten, welches beim VT benötigt wird, können die Matlabmodelle beinhalten.

Abschließend kann man sagen, dass Anbieter von Simulatoren und Testern im Bereich Virtueller Test weltweit zusammenarbeiten. Diesbezüglich sind einige interessante VT-Entwicklungsumgebungen entstanden, die versuchen, den Einsatz von virtuellen Testmethoden zu erleichtern.

Anhang D

MOSFET Level1-Grundlagen

In dem nachfolgenden Kapitel sollen die Grundlagen des MOSFET Level1-Modells, welches in der Arbeit verwendet wird, näher vorgestellt werden. [Tie99], [Che99] und [Gei90] sind hierbei als Quellen verwendet worden.

D.1 MOSFET Level1-Eigenschaften

Ein MOSFET Level1-Modell ist für integrierte MOSFETs in ihrer einfachsten Form gültig. Für MOSFETs, die z.B. als vertikale DMOS-FETs ausgeführt sind, zeigen teilweise ein anderes Verhalten. Hierfür gibt es dann Modelle höheren Levels (z.B. BSIM, EKV..). Im nachfolgenden Abschnitt soll das MOSFET Level1-Modell in seiner mathematischen Funktion näher gebracht werden.

Bei einem MOSFET-Modell gibt es drei unterschiedliche Arbeitsbereiche, die sich durch die Spannungsdifferenzen an Gate, Source und Drain definieren. Hierbei handelt es sich um den **Sperrbereich**, den **Linearen Bereich** und den **Sättigungsbereich**. In der Literatur haben sich auch noch andere Namen für die jeweiligen Bereiche etabliert, siehe Tabelle D.1.

Betriebsbereiche:

Bild D.1 zeigt das Ausgangskennlinienfeld eines NMOSFET-Transistors. Ist die Gate-Source-Spannung U_{GS} kleiner als die Schwellspannung U_{th}, kann sich kein leitender Kanal ausbilden. Damit befindet sich der Transistor im Sperrbereich, wo der Drainstrom $I_D \approx 0$ unabhängig von der Drain-Source-Spannung U_{DS} ist. Überschreitet U_{GS} die Schwellspannung U_{th}, bildet sich ein Kanal, in dem Strom fließen kann. Der Strom I_D ist für kleine U_{DS} näherungsweise proportional

Sperrbereich	$U_{GS} < U_{th}$
Linearer Bereich, aktiver Bereich, ohmscher Bereich, Triodenbereich	$U_{GS} \geq U_{th} \wedge 0 \leq U_{DS} < U_{DS,sat}$
Sättigungsbereich, Abschnürbereich, Pentodenbereich	$U_{GS} \geq U_{th} \wedge U_{DS} \geq U_{DS,sat}$

Tabelle D.1: Betriebsbereiche eines MOS-Modells

zu U_{DS}. Dieser Bereich wird daher als linearer Bereich bezeichnet. Wird U_{DS} über $U_{DS,sat} = U_{GS} - U_{th}$ erhöht, wird der Kanal auf der Drainseite abgeschnürt und I_D steigt nur noch sehr schwach an. Der Transistor befindet sich im Sättigungsbereich.

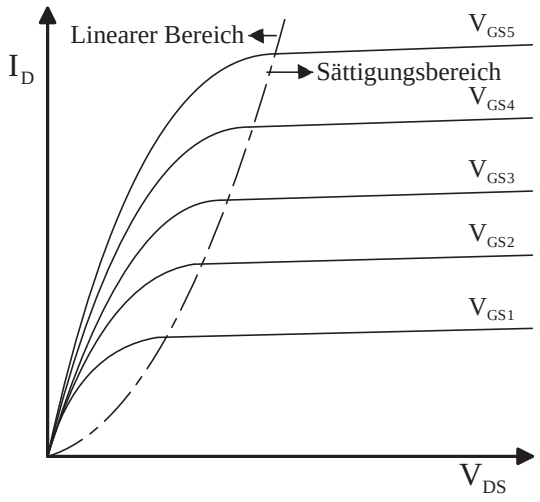

Abbildung D.1: Kennlinie eines NMOSFET-Transistors

Drainstrom

Der Drainstrom lässt sich basierend auf der Annahme, dass eine ideale Ladungsverteilung im Kanal herrscht, näherungsweise mit Sah's Modell [Che99] berechnen.

$$I_D = \begin{cases} 0 & U_{GS} < U_{th} \\ \frac{K_n * W}{L} * U_{DS} * (U_{GS} - U_{th} - \frac{U_{DS}}{2}) & U_{GS} \geq U_{th}; \\ & 0 \leq U_{DS} < U_{DS,ab} \\ \frac{K_n * W}{2 * L} * (U_{GS} - U_{th})^2 & U_{GS} \geq U_{th}, \\ & U_{DS} \geq U_{DS,ab} \end{cases} \tag{D.1}$$

Das schwache Ansteigen von I_{DS} im Sättigungsbereich wird durch die Kanallängenmodulation verursacht. Sie lässt sich mit dem Shichman-Hodges-Modell [Che99] beschreiben. Damit lauten die Gleichungen für einen n-Kanal-MOSFET in den verschiedenen Arbeitsbereichen wie folgt:

$$I_D = \begin{cases} 0 & U_{GS} < U_{th} \\ \frac{K_n * W}{L} * U_{DS} * (U_{GS} - U_{th} - \frac{U_{DS}}{2}) * (1 + \frac{U_{DS}}{U_A}) & U_{GS} \geq U_{th}; \\ & 0 \leq U_{DS} < U_{DS,ab} \\ \frac{K_n * W}{2 * L} * (U_{GS} - U_{th})^2 * (1 + \frac{U_{DS}}{U_A}) & U_{GS} \geq U_{th}, \\ & U_{DS} \geq U_{DS,ab} \end{cases} \tag{D.2}$$

Für ein p-Kanal-MOSFET gelten folgende Gleichungen:

$$I_D = \begin{cases} 0 & U_{GS} > U_{th} \\ -\frac{K_n * W}{L} * U_{DS} * (U_{GS} - U_{th} - \frac{U_{DS}}{2}) * (1 - \frac{U_{DS}}{U_A}) & U_{GS} \leq U_{th}; \\ & U_{DS,ab} < U_{DS} \leq 0 \\ -\frac{K_n * W}{2 * L} * (U_{GS} - U_{th})^2 * (1 - \frac{U_{DS}}{U_A}) & U_{GS} \leq U_{th}, \\ & U_{DS} \leq U_{DS,ab} \end{cases} \tag{D.3}$$

mit $U_{DS,ab} = U_{GS} - Uth$

Der Steilheitskoeffizient oder auch Transkonduktanz-Koeffizient K_n bzw K_p ist ein Maß für die Steigung der Übertragungskennlinie eines MOSFETs und lässt sich wie folgt berechnen:

$$K_n = \frac{\mu_n * \varepsilon_0 * \varepsilon_{r,ox}}{d_{ox}} \tag{D.4}$$

$$K_p = \frac{\mu_p * \varepsilon_0 * \varepsilon_{r,ox}}{d_{ox}} \tag{D.5}$$

dabei ist μn bzw. μp die Beweglichkeit der Ladungsträger im Kanal.

$\mu n = 0,05 \ldots 0,07 \frac{m^2}{Vs}$
$\mu p = 0,015 \ldots 0,025 \frac{m^2}{Vs}$.

Der Drainstrom in Abhängigkeit von der Drain-Source-Spannung wird im Sperrbereich durch die **Kanallängenmodulation** verursacht. Beim Extrapolieren der I_D-U_{DS}-Kennlinien (für verschiedene U_{GS}) schneiden sich alle Kennlinien näherungsweise in einem Punkt, siehe Bild D.2. Dieser Punkt wird in Anlehnung an den Bipolartransistor **Early Spannung** *UA* genannt.

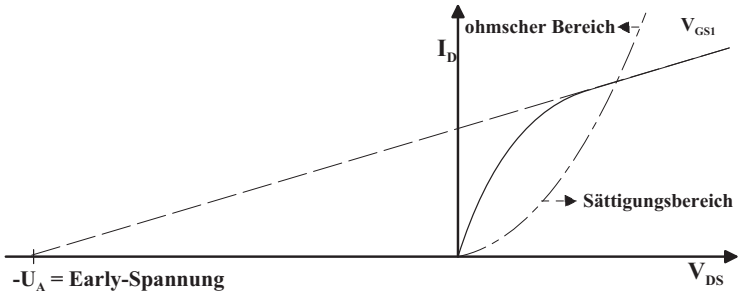

Abbildung D.2: Early Spannung

Die Early-Spannung wird oft durch den Kanallängenmodulationsparameter λ beschrieben:

$$\lambda = \frac{1}{U_A} \tag{D.6}$$

Schwellenspannung

Die Gate-Soucre-Spannung, ab der sich unter dem Gate ein Kanal bildet, welcher zum Stromfluss zwischen Drain und Source führt, wird Schwellenspannung U_{th} genannt. Dies ist auch der

Übergang zwischen dem Sperrbereich und dem Linearen Bereich. Die Spannung lässt sich nach folgenden Formeln berechnen:

n-Kanal:

$$U_{th} = U_{th,0} + \gamma(\sqrt{U_{inv} - U_{BS}} - \sqrt{U_{inv}}) \tag{D.7}$$

p-Kanal:

$$U_{th} = U_{th,0} - \gamma(\sqrt{U_{inv} - U_{BS}} - \sqrt{U_{inv}}) \tag{D.8}$$

Wie aus der Formel ersichtlich ist, hängt die Schwellenspannung von der Bulk-Source-Spannung U_{BS} ab (Substrat-Effekt). Null-Schwellspannung $U_{th,0}$ und Substrat-Steuerfaktor $\gamma \approx 0.3 \ldots 0.8\sqrt{V}$ sind Prozessparameter. Die Inversionsspannung $U_{inv} \approx 0.55 \ldots 0.8$ V hängt von der Substrat-Dotierungsdichte N_{sub} und der Oxiddichte d_{ox} ab, und lässt sich nach folgender Beziehung herleiten.

$$\gamma = \frac{\sqrt{2q * \varepsilon_{r,Si} * N_{sub}}}{\varepsilon_0} * \frac{d_{ox}}{\varepsilon_{r,ox}} \tag{D.9}$$

$$Uinv = 2 * U_T * \ln \frac{N_{sub}}{n_i} \tag{D.10}$$

Substratdioden

Wie in Bild D.3 ersichtlich ist, besteht zwischen Bulk / Drain und Bulk / Source ein PN-Übergang. Diese Übergänge haben ein Diodenverhalten und werden als Substrat-Dioden bezeichnet. Folgende Gleichungen sind hierfür gültig:

$$I_{D,S} = I_{S,S} * (e^{\frac{U_{BS}}{n * U_t}} - 1) \tag{D.11}$$

$$I_{D,D} = I_{S,D} * (e^{\frac{U_{BD}}{n * U_t}} - 1) \tag{D.12}$$

Sättigungssperrströme $I_{S,S}$ und $I_{S,D}$

Emissionsfaktor n ≈ 1

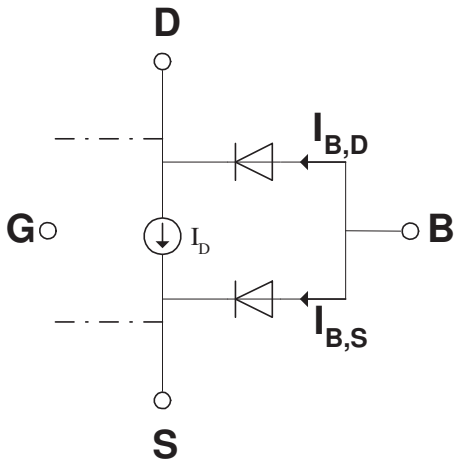

Abbildung D.3: Substrat-Dioden

Bahnwiderstände

Die Source, Drain, Gate und Bulk Anschlüsse verfügen über Bahnwiderstände, die sich aus dem Kontaktwiderstand der Metallisierung und dem Widerstand des jeweiligen Gebietes zusammensetzen. Die Bahnwiderstände können direkt oder unter Verwendung des Schichtwiderstandes R_{sh} und den Multiplikatoren $n_{RG}, n_{RS}, n_{RD}, n_{RB}$ verwendet werden.
Bei integrierten Schaltungen ist der Schichtwiderstand eine Eigenschaft des MOS-Prozesses. Typische Werte für einen n-Kanal-MOSFET liegen bei $R_{sh} \approx 20 \ldots 50\Omega$ und für einen p-Kanal-MOSFET bei $R_{sh} \approx 50 \ldots 100\Omega$.

$$\begin{pmatrix} R_G \\ R_S \\ R_D \\ R_B \end{pmatrix} = R_{sh} * \begin{pmatrix} n_{RG} \\ n_{RS} \\ n_{RD} \\ n_{RB} \end{pmatrix} \tag{D.13}$$

Dynamisches Verhalten

Für die „Transiente“ Simualtion ist bei Ansteuerung des MOSFETs mit pulsförmigen Signalen das „dynamische Verhalten"´ zu beobachten.

Hierfür sind Kapazitäten zwischen den verschiedenen Bereichen eines MOSFETs verantwortlich (siehe Bild D.4), die sich in drei verschiedene Gruppen aufteilen lassen.

- Kanalkapazitäten

- Überlappungskapazitäten

- Sperrschichtkapazitäten

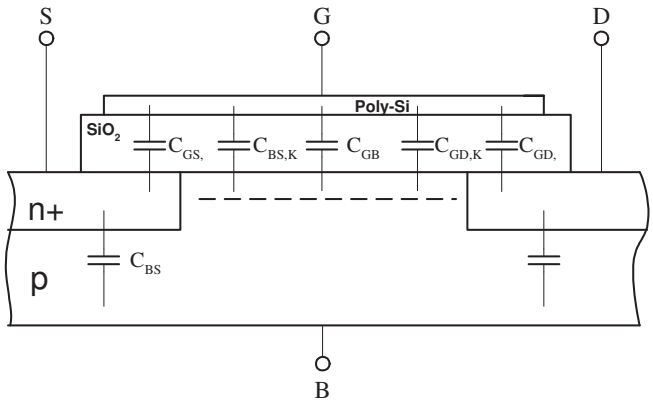

Abbildung D.4: Kapazitäten eines MOSFETs

Kanalkapazitäten

Die Kanalkapazitäten $C_{GS,K}$ und $C_{GD,K}$ sind nur wirksam, wenn der MOSFET leitet. Sie beschreiben die kapazitive Wirkung zwischen dem Gate und dem Kanal. Ist kein Kanal vorhanden, das heißt der MOSFET sperrt, erhält man eine Kapazität $C_{GB,K}$ zwischen Gate und Bulk.
Elektrisch gesehen bildet das Gate mit dem darunter liegenden Kanal einen Plattenkondensator mit der Oxidkapazität C_{ox}.

$$C_{ox} = \varepsilon_{ox} * \frac{A}{d_{ox}} = \varepsilon_0 * \varepsilon_{r,ox} * \frac{W * L}{d_{ox}} \tag{D.14}$$

Wenn sich kein Kanal bildet, das heisst der MOSFET ist im **Sättigungsbereich**, wirken die Kapazitäten zwischen Gate und Bulk.

In der Literatur haben sich einige Modellansätze entwickelt, die mehr oder weniger komplex sind (z.B. Berücksichtigung von Kurzkanaleffekten). Hierzu zählen beispielsweise das Lumped Modell [Che99], Meyer Model [Cad07], Yang-Chatterjee Modell [Cad07], oder BSIM Charge Modell [Cad07].

$$\left.\begin{aligned} C_{GS,K} &= 0 \\ C_{GD,K} &= 0 \\ C_{GB,K} &= C_{ox} \end{aligned}\right\} \quad \text{für } U_{GS} < U_{th} \tag{D.15}$$

Im **linearen Bereich** hängen die Kapazitäten von U_{DS} und U_{GS} ab. Entsprechend der Ladungsverteilung im Kanal vom Source- bis zum Drain-Gebiet teilt sich die Oxidkapazität näherungsweise nach folgender Formel auf:

$$\left.\begin{aligned} C_{GS,K} &= \frac{2}{3} * C_{ox} * \left(1 - \left(\frac{U_{GS} - U_{th} - U_{DS}}{2 * (U_{GS} - U_{th}) - UDS}\right)^2\right) \\ C_{GD,K} &= \frac{2}{3} * C_{ox} * \left(1 - \left(\frac{U_{GS} - U_{th}}{2 * (U_{GS} - U_{th}) - UDS}\right)^2\right) \\ C_{GB,K} &= 0 \end{aligned}\right\} \quad \text{für } U_{GS} \geq U_{th}, U_{DS} < U_{GS} - U_{th} \tag{D.16}$$

Im Sättigungsbereich besteht keine Verbindung mehr zwischen dem Kanal und dem Drain-Gebiet. Damit wirkt nur noch die kapazitive Wirkung zwischen Gate und Source als Kanalkapazität.

$$\left.\begin{aligned} C_{GS,K} &= \frac{2}{3} * C_{ox} \\ C_{GD,K} &= 0 \\ C_{GB,K} &= 0 \end{aligned}\right\} \quad \text{für } U_{GS} \geq U_{th}, U_{DS} \geq U_{GS} - U_{th} \tag{D.17}$$

Die hier beschriebene Berechnung der Kanalkapazitäten wird im MOSFET Level1-Modell verwendet. Problematisch ist hier der Sprung der Gesamtkapazität vom z.B. Sperrbereich in den Sättigungsbereich (von Cox auf $\frac{2}{3} * Cox$). Der Übergang ist bei einem realen MOSFET stetig. In der nachfolgenden Bild ist der Kapazitätsverlauf in den verschiedenen Bereichen dargestellt.

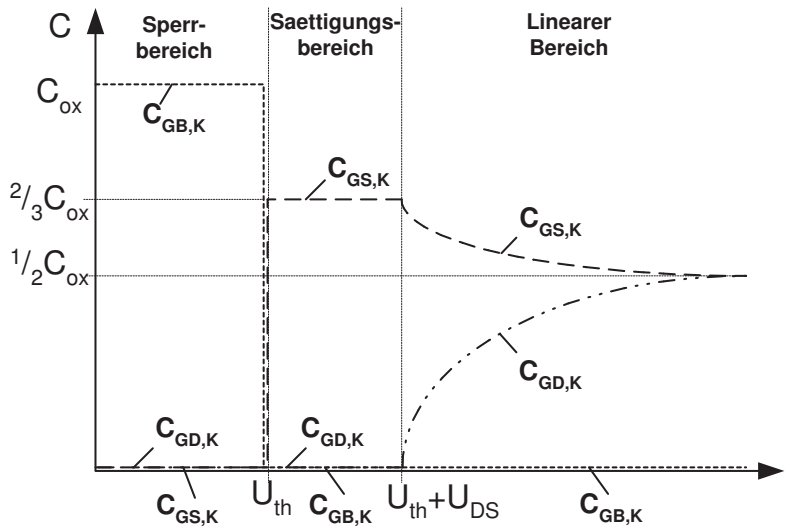

Abbildung D.5: Spannungsabhängigkeit der Kanalkapazitäten eines MOSFETs

Überlappungskapazitäten

Überlappungskapazitäten entstehen durch Überlappung an den Rändern vom Gate-Gebiet zum Kanal. Zur Berechnung werden die randlängenbezogenen Kapazitätsbelege, die durch Messung ermittelt werden können, herangezogen.

$$C_{GS,o} = C'_{GS,o} * W \tag{D.18}$$

$$C_{GD,o} = C'_{GD,o} * W \tag{D.19}$$

$$C_{GB,o} = C'_{GB,o} * L \tag{D.20}$$

Sperrschichtkapazitäten

Ein pn-Übergang besitzt eine spannungsabhängige Sperrschichtkapazität. Diese ist abhängig von der Dotierung der aneinander grenzenden Gebiete, der Fläche des pn-Übergangs sowie der anliegenden Spannung. Bei einem MOSFET befinden sich diese pn-Übergänge zwischen Bulk und Source bzw. zwischen Bulk und Drain.

$$C_{BS} = \frac{C_{S0,S}}{(1-\frac{U_{BS}}{U_{Diff}})^{ms}} \quad \text{für } U_{BS} \leq 0 \tag{D.21}$$

$$C_{BD} = \frac{C_{S0,D}}{(1-\frac{U_{BD}}{U_{Diff}})^{ms}} \quad \text{für } U_{BD} \leq 0 \tag{D.22}$$

Nullkapazitäten $C_{S0,S}$ und $C_{S0,D}$
Diffusionsspannung U_{Diff}
Kapazitätskoeffizienten $ms \approx \frac{1}{3} \ldots \frac{1}{2}$

Effekte, die keine Berücksichtigung im Level1-Modell finden

- Kurzkanal-Effekt
- Schmalkanal-Effekt
- Unterschwellen-Effekt

Kurzkanal-Effekt

Der Bereich unter dem Kanal wird bei geringen Kanallängen von den Sperrschichten der Bulk-Source-Diode und der Bulk-Drain-Diode stark eingeengt. In diesem Fall findet eine zunehmende Kompensation der dort herrschenden Raumladung durch Landungen im Source- und Draingebiet statt. Dieses Verhalten führt zu einer Reduzierung der Gateladung und somit nimmt die Schwellspannung U_{th} ab und der Drainstrom I_D nimmt dementsprechend zu.

D.2 Ermittlung der MOSFET Level1-Parameter

Die originale Ausgangstreiberstufe verwendet Transistormodelle, die z.B. mit einer EKV-Modellkarte realisiert sind. Die erste Aufgabe besteht darin, die Parameter für ein MOSFET Level1-Modell zu bestimmen. Manche Parameter können direkt vom EKV-Modell übernommen werden, sowie beispielsweise U_{tho}, t_{ox}, l_d, r_d, r_s, oder c_j. Andere können nur durch Simulationen bestimmt werden, z.B. K_p, U_{th}, γ, U_{inv} und λ. Besteht das originale Transistormodell nicht nur aus einer Modellkarte, sondern hat beispielsweise noch externe Komponenten wie Dioden, spannungsgesteuerte Widerstände oder spannungsgesteuerte Kapazitäten, so müssen die dahinter stehenden Level1 Parameter ebenfalls per Simulation ermittelt werden. Hier ist keine genaue Ermittlung

der Parameter möglich, weil diese arbeitspunktabhängig sind. Für die meisten Anforderungen eines Verhaltensmodells sind diese aber ausreichend genau.

Eine weitere Ungenauigkeit bei der Ermittlung der MOSFET Level1-Parameter stellen verschiedene Dimensionierungen eines gleichen Transistortyps, der z.B. mit einer EKV-Modellkarte realisiert ist, dar. Es würden sich bei unterschiedlichen Dimensionen unterschiedliche Parameter ergeben. Der Grund hierfür sind unterschiedliche Berechnungsgrundlagen, die sich nicht ohne weiteres ineinander überführen lassen [Cad07], [Xun04].

Die genaueste, aber eine sehr zeitintensive Methode wäre, für jede Dimensionierung die Parameter zu ermitteln. Ein anderer Vorschlag ist das Einführen von Dimensionierungsklassen, in dem Bereiche festgelegt werden, und nur für diese Bereiche eine Ermittlung stattfindet.
Im nachfolgenden Abschnitt soll kurz auf die Ermittlung der Parameter mit Hilfe einer Simulation eingegangen werden.

Ermittlung von λ

Zur Ermittlung von λ werden zwei Punkte der DC-MOS-Kennlinie im Sperrbereich benötigt, siehe Bild D.6.

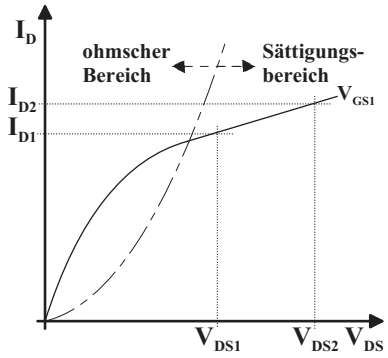

Abbildung D.6: Ermittlung von λ aus einer DC-OS-Kennlinie [Gei90]

Nach folgender Formel lässt sich daraus λ berechnen:

$$\lambda = \frac{I_{D2} - I_{D1}}{I_{D1} * U_{D2} - I_{D2} * U_{D1}} \tag{D.23}$$

Ermittlung von K_p bzw. K_n

Der Steilheitskoeffizient K_p oder auch K_n kann ebenfalls durch die DC- Kennlinie ermittelt werden. I_D und U_{DS} werden bei einem festen U_{GS} in der Kennlinie ermittelt. Mit diesen Werten kann Kp durch folgende Gleichung berechnet werden:

$$K_p = \frac{2 * L * I_D}{W * (U_{GS} - U_{th})^2 * (1 + \lambda * U_{DS})} \tag{D.24}$$

Ermittlung von U_{th}
Mit festen U_{DS} und U_{BS}, zwei unterschiedlichen U_{GS} und der zugehörigen I_D im Sperrbereich lässt sich U_{th} nach folgender Formel berechnen:

$$U_{th} = \frac{U_{GS2} * (I_{D1}/I_{D2})^{0.5} - U_{GS1}}{(I_{D1}/I_{D2})^{0.5} - 1} \tag{D.25}$$

Ermittlung von γ und U_{inv}
Die Parameter γ und U_{inv} lassen sich basierend auf der Formel D.7 mit zwei unterschiedlichen U_{BS} und den dazu ermittelten U_{th} (siehe Formel D.25) berechnen.

Ermittlung der Kapazitäten
Bei der Ermittlung der einzelnen Kapazitäten wie $C_{GS,K}$, $C_{GB,K}$, $C_{GD,K}$, C_{BS} und C_{BD} wird eine AC- Simulation zur Hilfe genommen. Die AC-Quelle wird hierbei zwischen den jeweiligen zwei Klemmen der zu ermittelnden Kapazität angeschlossen (siehe Bild D.7). Ebenso wie bei den

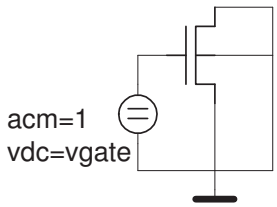

Abbildung D.7: Ermittlung der Kapazitäten eines MOS-Transistors

DC-Parametern gilt auch hier, dass je nach Art des Transistormodells die Ermittlung der Parameter eine Abhängigkeit vom Arbeitspunkt und der Dimensionierung aufweisen, die nicht mit den Berechnungen wie in Gleichung D.15 , D.16 oder D.17 übereinstimmen. Tritt dieser Fall ein, muss C_{ox} in dem zu erwartenden Arbeitspunkt beim Einsatz in der Schaltung ermittelt werden. Die Ungenauigkeiten, die sich hierbei ergeben, sind für ein Verhaltensmodell akzeptabel.

D.3 Spectre MOSFET Level1-Modell

Die zuvor beschriebenen Eigenschaften finden in fast allen Simulatoren ihre Anwendung. Der in dieser Arbeit verwendete Simulator ist Spectre bzw. AMS-Designer von Cadence, welcher

ebenso ein Level1-Modell als Simulator-Primitive zur Verfügung stellt. Dieses Modell wird als Referenz für das variable MOSFET HDL-Modell verwendet.

Beim Spectre MOSFET Level1-Modell ist es möglich die Modell-Größen in skalierbarer oder

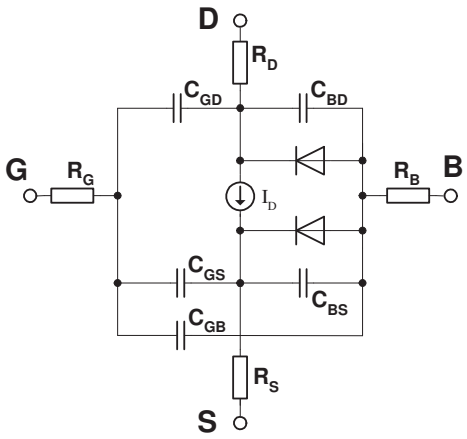

Abbildung D.8: Blockschaltbild MOSFET Level1

in effektiver Form anzugeben. Ein Beispiel hierfür sind die Bahnwiderstände. Diese können mit $n_{RG} \ldots n_{RB}$ und R_{sh} skalierbar oder mit $R_G \ldots R_B$ effektiv angegeben werden. Werden im Modell beide Formen übergeben, hat die effektive Form Vorrang.

Ebenso ist es mit berechneten Werten, wie bei den Parametern K_n und γ. Diese können mit den Parametern d_{ox}, μ_n, U_{inv} und N_{sub} berechnet (siehe Gleichung D.4) oder direkt angegeben werden. Bei widersprüchlichen Angaben hat die direkte Angabe Vorrang vor dem berechneten Wert. In Bild D.8 ist das Blockschaltbild eines Level1-Modells dargestellt.

Anhang E

Bewertungsmethoden

Methoden zur Bewertung erzielter Ergebnisse finden sich nicht nur in den Naturwissenschaften wieder, sondern auch in vielen anderen Bereichen wie z.B. Sozialwissenschaften, Biologie, Wirtschaftswissenschaften oder Marktforschungen. Selbst in dem Bereich der Naturwissenschaften gibt es eine Vielzahl von Gebieten, in denen Bewertungsmethoden ihren Einsatz finden. Ebenso gibt es eine große Anzahl von Bewertungskriterien. In dieser Arbeit werden als Kriterien die Genauigkeit des Modells und der Performancegewinn zur Originalschaltung zu Grunde gelegt. Im nachfolgenden Kapitel werden diese näher beschrieben.

E.1 Modellgenauigkeit

Berechnungen von Genauigkeiten finden in der Literatur, wie in vielen anderen Bereichen ihren Einsatz. Die Bildverarbeitung dürfte hierbei eines der bekanntesten Gebiete darstellen. Nachfolgend sollen Methoden zur Berechnung der Genauigkeiten einzelner Verhaltensmodelle zur Original-Transistorschaltung vorgestellt werden. Im folgenden soll $\vec{y}(t)$ die Referenzkurve im Intervall $t \in [a,b]$ mit dem Signal $y(t)$ des Verhaltensmodells darstellen. Die Genauigkeit lässt sich allgemein in Abhängigkeit der verschiedenen Fehlernormen $||a_{...}||$ und der Abstandsmaße a_m wie folgt darstellen:

$$d_p^m(y,\vec{y}) = ||a_m||_f \tag{E.1}$$

E.1.1 Abstandsmaße

Die Berechnung des Abstandes bzw. der Distanz zwischen zwei Kurven können durch verschiedene Distanzfunktionen durchgeführt werden, hierzu zählen beispielsweise:

- Hamming-Distanz

- Lp-Distanz oder Minkowski-Metrik
- City-Block-Distanz, Manhattan-Distanz
- Maximum- bzw. Tschebyschow-Distanz
- Euklidische Distanz
- Canberra-Distanz
- Mahalanobis-Distanz

Distanzfunktionen beschreiben den Grad der Übereinstimmung von Vektoren und werden oft auch als Metriken bezeichnet. In dieser Arbeit wird die Euklidische Distanz und die Vertikale Distanz (Ordinatendifferenz) verwendet, die für einen zweidimensionalen Raum gut geeignet sind.

Vertikale Distanz

Die Vertikale Distanz beschreibt die absolute Differenz der Ordinaten der Referenzkurve und der Modellkurve (Ordinatendifferenz). Der Anstand lässt sich wie folgt berechnen:

$$z_V(t) = y(t) - \widehat{y}(t) \tag{E.2}$$

Wobei hier $y(t)$ die Kurve des Verhaltensmodells und $\widehat{y}(t)$ die Referenzkurve im Intervall $t \in [a,b]$ darstellt.

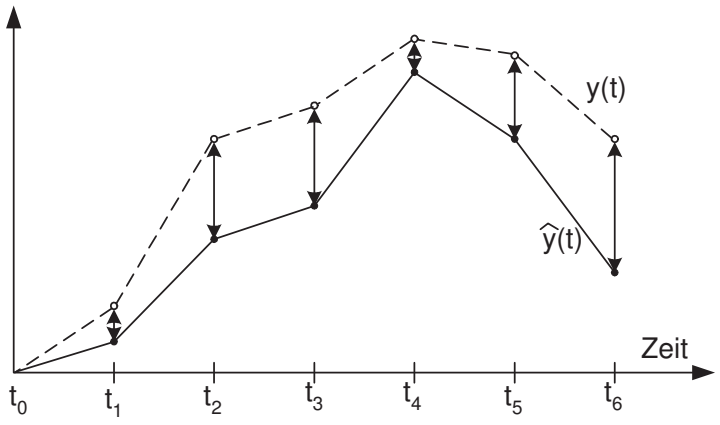

Abbildung E.1: Vertikaler Distanz

Vorteil dieser Funktion ist die einfache und schnelle Berechnung des Abstandes, jedoch bei Auftreten von Spikes, kann diese Methode eine unerwartet hohe Ungenauigkeit verursachen (siehe Bild E.1). Ein weiterer Nachteil wird bei den zeitlich versetzten Sprungantworten sichtbar, wie

es z.B. bei An- und Abstiegsflanken der Fall ist. In diesen Bereichen würde eine horizontale Abstandsfunktion sinnvoller sein.

Euklidische Distanz
Die Euklidische Distanz ist die allgemeine mathematische Beschreibung des direkten Abstandes zwischen zwei Vektoren und wird wie folgt definiert:

$$z_E(x,y) = \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2} \tag{E.3}$$

Ebenso wird der kürzeste Abstand eines Punktes der Referenzkurve zur Modellkurve als Euklidische Distanz bezeichnet (siehe Bild E.1).

$$z_E(t_i) = \min_{t\in[a,b]} \sqrt{(y(t) - \widehat{y}(t))^2 + (t - t_i)^2} \tag{E.4}$$

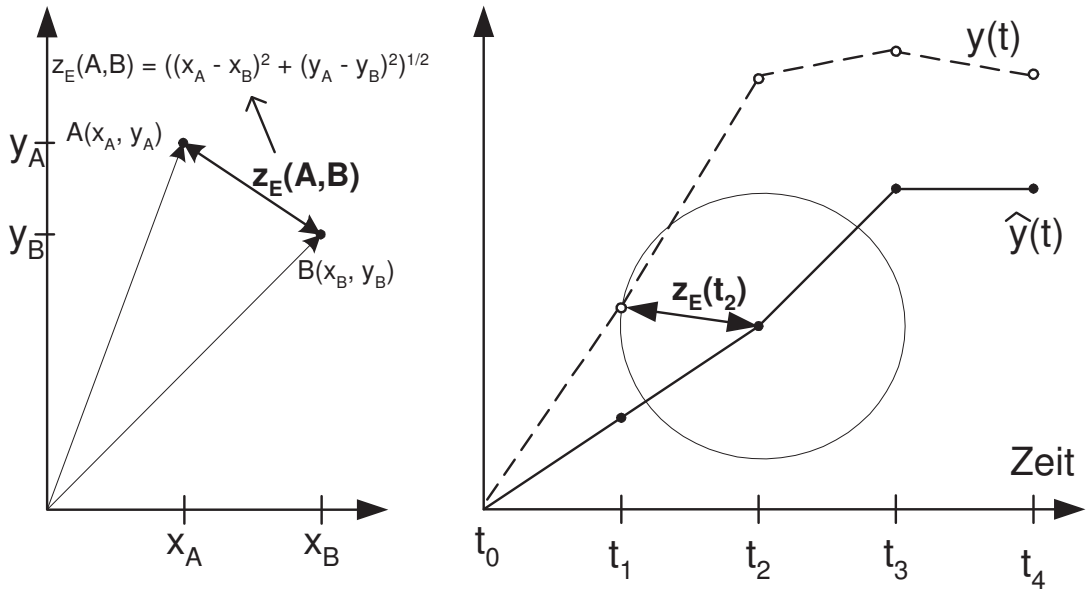

Abbildung E.2: Euklidische Distanz

In der Bild E.2 ist ein Beispiel für die Berechnung des minimalen Abstandes zum Zeitpunkt t_2

von der Referenzkurve zur Modellkurve dargestellt. Der Radius des Kreises ist für den Punkt $y(t_2)$ die Euklidische Distanz.

Bei der Berechnung der minimalen Distanz, werden sowohl die Werte der Abszisse als auch der Ordinate verwendet. Da es sich hierbei um elektrische Größen (z.B. Strom, Spannung) für die Ordinaten und die Größe Zeit für die Abszisse handelt, und diese in unterschiedlichen Größenverhältnissen vorliegen können, muss ein Skalierungsfaktor s mit berücksichtigt werden. Hieraus ergibt sich folgende Modifikation:

$$z_E(t_i) = \min_{t\in[a,b]} \sqrt{(y(t)-\widehat{y}(t))^2 + s^2(t-t_i)^2} \tag{E.5}$$

Die Einheitsgrößen der x- und y-Achse sind Variable und können immer wieder unterschiedlich definiert werden. So kann beispielsweise eine Stromkurve eine deutlich andere Größe als eine Spannungskurve besitzen. Beide sollen aber den gleichen Einfluss auf die Zeitachse bei der Berechnung des minimalen Abstandes beitragen. Um dieses Problem zu lösen wird der Skalierungsfaktor wie folgt berechnet:

$$s = \frac{\Delta x}{\Delta y} \tag{E.6}$$

dabei gilt:

$$\Delta x = t_{end} - t_{start} \tag{E.7}$$

$$\Delta y = y_{end} - y_{start} \tag{E.8}$$

Um mehr Flexibilität für die Berechnung der Distanzen zu bekommen, ist ein Start- und End-Zeitpunkt (t_{start}, t_{end}) zu definieren. Mit dieser Methode kann beispielsweise gezielt Anstiegsflanken, deren Anstiegszeit im Verhältnis zur gesamten Simulationszeit sehr gering sein kann, untersucht werden. Ebenso wird für die y-Achse ein Start- und Endwert(y_{start}, y_{end}) erwartet. Das nachfolgende Beispiel soll die Berechnung des Skalierungsfaktors verdeutlichen.

Die Euklidische Distanz wirkt wie ein Tiefpass und filtert somit Spikes, die bei der Simulation auftreten können, heraus. Dies ist ein entscheidender Vorteil gegenüber der Vertikalen Distanz. Es wird immer der minimale Abstand von einem Punkt der Referenzkurve zur Modellkurve berechnet. Ein Nachteil dieser Berechnungsmethode ist der Rechenaufwand. Er ist im Gegensatz zur vertikalen Distanz um ein Vielfaches höher und führt somit zu längeren Berechnungszeiten. Für die Laufzeit der Simulation ist dieser Aufwand unerheblich, da die Bewertung und somit die Bestimmung der Ungenauigkeit nur bei der Charakterisierung auftreten.

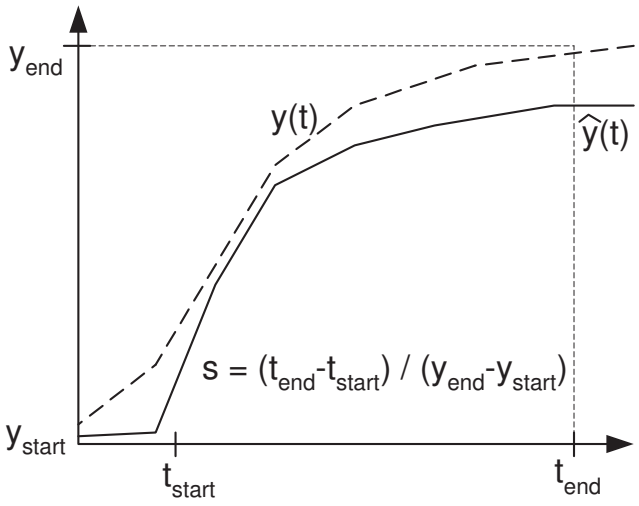

Abbildung E.3: Berechnung des Skalierungsfaktors

E.1.2 Fehlernormen

Mit den Abstandsmaßen allein, lässt sich noch keine Aussage über das Maß der Güte der Verhaltensmodelle machen. Hier werden noch Fehlernormen benötigt, die festlegen, von den Differenzen zwischen Referenzkurve und Modellkurve zu Größe des Fehlers und somit zur Genauigkeit des Verhaltensmodells zu kommen. In der hier angewandten Fehlernorm werden alle Distanzen in dem zu untersuchenden Intervall aufsummiert und durch die Gesamtfläche, die durch y_{start} und y_{end} definiert ist, dividiert.

$$||z||_1 = \frac{\int_{t=t_{start}}^{t_{end}} |z(t)|dt}{\int_{t=t_{start}}^{t_{end}} (y_{end}(t) - y_{start}(t))} \tag{E.9}$$

Vorteil dieser Norm ist die flexible Berechnung der Genauigkeit durch die Definition der Bereiche sowohl in x als auch in y Richtung. Eine weitere verwendete Fehlernorm richtet sich an dem maximalen Fehler, der in dem zu untersuchenden Intervall auftritt. Hier wird das Verhältnis der Summe der Distanzen zu jedem Auswertungszeitpunkt zur größten Distanz, die für die Modellkurve vorkommt, gebildet.

$$||z||_2 = \int_{t=t_{start}}^{t_{end}} \frac{|z(t)|}{\max_{t\in[t_{start},t_{end}]} |(z_t(t)|)} dt \tag{E.10}$$

Die hier vorgestellten Fehlernormen sollen in dem nachfolgenden Bild veranschaulicht werden.

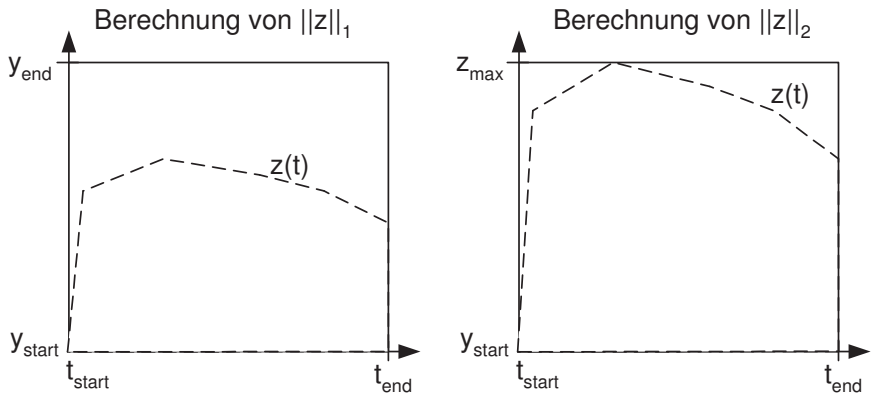

Abbildung E.4: Fehlernormen

E.2 Simulations-Ergebnis-Analyse-Programm (SEAP)

In Rahmen dieser Arbeit ist ein Programm zur Analyse der Genauigkeit einer oder mehrerer Modellkurven gegenüber einer Referenzkurve entstanden. Als Basis für die Berechnungen sind die zuvor beschriebenen Methoden zur vertikalen und zur minimalen (Euklidischen) Distanz, ebenso wie die Fehlernormen mit eingeflossen. Das Programm ist in Matlab realisiert und wird zur Charakterisierung der verschiedenen Abstraktionsgrade bzw. Varianten der Verhaltensmodelle eingesetzt. Zur benutzerfreundlichen Bedienung ist im Zuge dieser Programmentwicklung ebenfalls ein graphische Benutzeroberfläche entstanden. Dieses Hilfsmittel wird im nachfolgenden Abschnitt näher beschrieben.

Allgemeine Beschreibung der Funktionalität:
SEAP ist ein Werkzeug zur Analyse der Simulationsergebnisse mit dem Ziel, die Modellgenauigkeit eines oder mehrerer Modelle zu bestimmen und qualitativ bewerten zu können. Als Eingabe benötigt SEAP die Ergebnisse einer Simulation in Form einer vom Simulator ausgegebenen CSV-Datei. Diese beschreibt in einer Art von Wertetabelle die Signalverläufe aller ausgewerteten Ausgangssignale über einer Menge von Auswertungszeitpunkten.

Aus den ersten vier Ausgangssignalen der eingelesenen CSV-Datei, kann der Benutzer eines der Signale als Referenzsignal auswählen, die übrigen werden als Modellsignale angenommen und auf ihre Abweichung hin untersucht. Falls der Benutzer nicht explizit ein Referenzsignal auswählt, legt SEAP standardmäßig das erste Ausgangssignal als Referenzsignal fest.

Zur Bestimmung der Abweichung der Modellsignale vom Referenzsignal berechnet SEAP für jedes der Modellsignale eine Distanz zum Referenzsignal. Von den verschiedenen Möglichkeiten, die für die Bestimmung der Distanz zweier Funktionen existieren, sind in SEAP zwei

realisiert: die vertikale Distanz und die minimale Distanz vom Referenzsignal zum Modellsignal. Standardmäßig geht SEAP von einer Bildung der vertikalen Distanz aus.

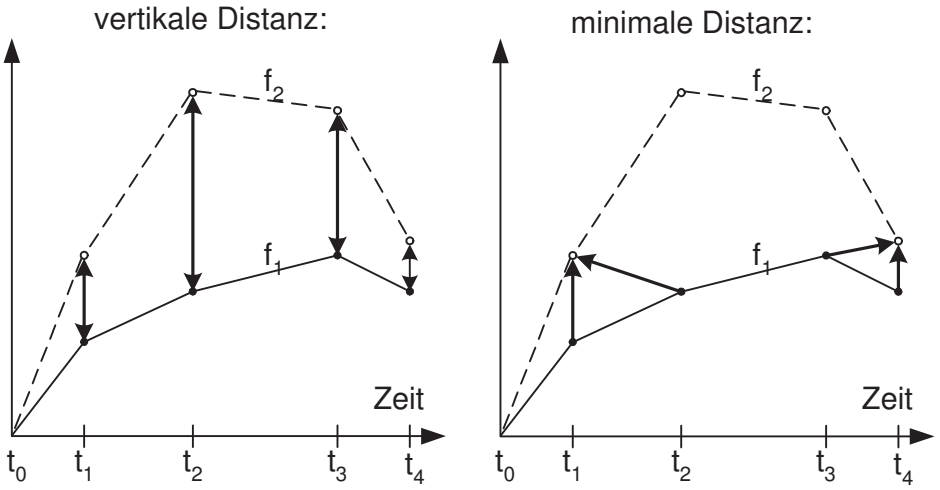

Abbildung E.5: Die vertikale/minimale Distanz

Bild E.5 stellt beide Distanzen am Beispiel zweier Funktionen f_1 und f_2 zu den Auswertungszeitpunkten t_0, t_1, t_2, t_3 und t_4 dar. Dabei sei f_1 das Referenzsignal und f_2 das Modellsignal. Die Bestimmung der jeweiligen Distanz für alle Auswertungszeitpunkte der gegebenen Signale liefert für jede der Modellfunktionen eine zugehörige Distanzfunktion. Die Funktionen der Distanzen zwischen den Modellsignalen und dem Referenzsignal werden von SEAP graphisch ausgegeben. Für das Beispiel aus Bild E.5, ist die graphische Ausgabe der vertikalen bzw. minimalen Distanzfunktion in Bild E.6 dargestellt.

Nach der Berechnung der Distanzen bestimmt SEAP für jede der gewonnenen Distanzfunktionen einen Normierungswert. Dieser bezeichnet das Verhältnis der Summe der Distanzen zu jedem Auswertungszeitpunkt zur größten Distanz, die für eines der Modellsignale vorkommt. Mit Hilfe dieses Normierungswertes können die Abweichungen der Modellsignale vom Referenzsignal nicht nur graphisch, sondern auch zahlenmäßig bewertet werden.

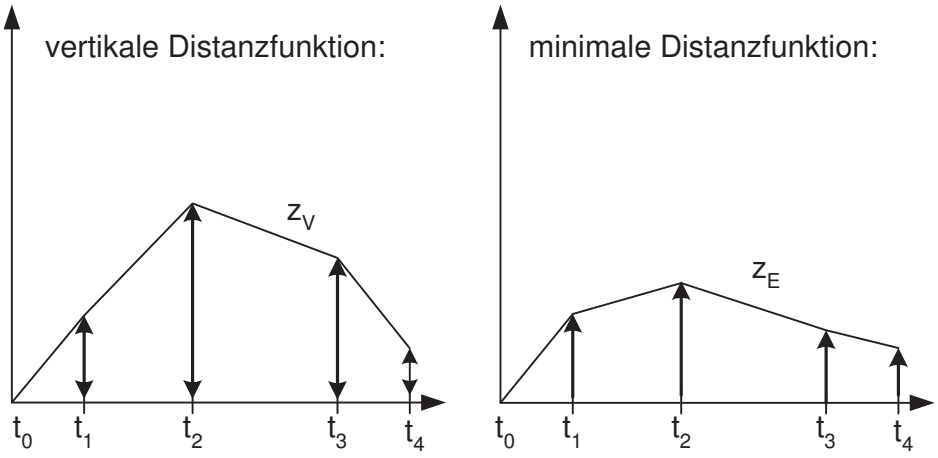

Abbildung E.6: Die Funktion der vertikalen/minimalen Distanz

Um die Genauigkeit bei der Berechnung der Distanzen zu erhöhen, hat der Benutzer die Möglichkeit, einen Verfeinerungsfaktor für die Ausgangssignale des Simulators einzustellen. Dieser Verfeinerungsfaktor ist eine natürliche Zahl größer Null und legt fest, in wie viele äquidistante Teilintervalle der Zeitraum zwischen zwei benachbarten Auswertungszeitpunkten unterteilt werden soll. Für jeden der neu hinzu gekommenen Auswertungszeitpunkte werden dann, wie in Bild E.7 für das Beispiel aus Bild E.5 und den Verfeinerungsfaktor 2 dargestellt, die Werte der Ausgangssignale interpoliert.

Anschließend wird unter Verwendung der hinzugekommenen interpolierten Signalwerte die jeweilige Distanz zwischen Referenzsignal und Modellsignal berechnet. Bild E.8 zeigt die Distanzfunktionen für das Beispiel aus Bild E.5 mit einer Verfeinerung um den Faktor 2.

Die Vergrößerung der Wertemenge führt sowohl bei der Bildung der vertikalen Distanz, als auch besonders bei der Bildung der minimalen Distanz, zu einer höheren Genauigkeit der Bewertung. Allerdings nimmt man damit eine höhere Berechnungsdauer in Kauf. Im Allgemeinen ist eine Verfeinerung der Wertemengen nicht zwingend notwendig, um die unterschiedlichen Abweichungen der Modellsignale qualitativ bewerten zu können. SEAP geht daher standardmäßig vom Verfeinerungsfaktor 1 aus.
Die Menge der Auswertungszeitpunkte, zu denen der Simulator die Werte der Ausgangssignale bestimmt hat, lässt sich vom Benutzer auf ein frei wählbares Teilintervall einschränken. Dadurch kann der Benutzer eine unter Umständen langwierige Berechnung durch Einschränkung der Menge der Auswertungszeitpunkte auf ein möglicherweise für die Unterscheidung der Ab-

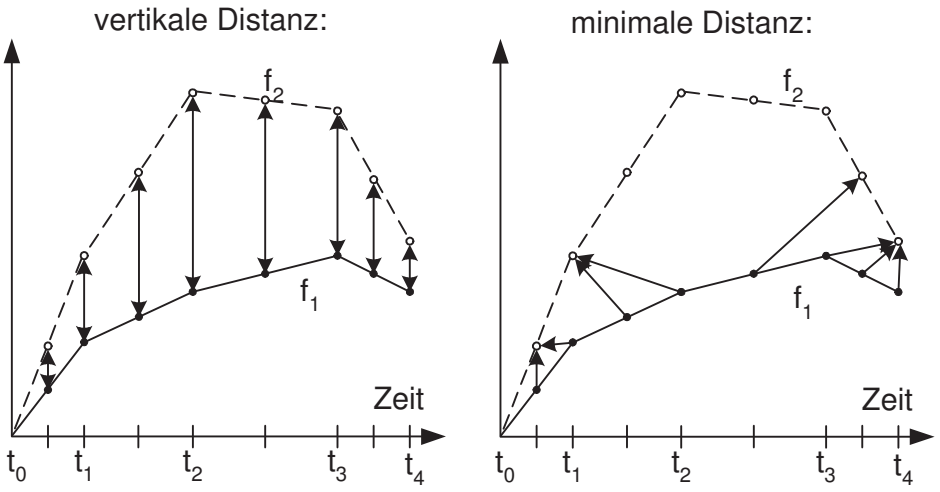

Abbildung E.7: Die vertikale/minimale Distanz mit Faktor 2

weichungen ausschlaggebendes Teilintervall mitunter wesentlich beschleunigen. Standardmäßig geht SEAP vom Vollbild aus, d.h. dass der kleinste Auswertungszeitpunkt der Simulation die linke und der größte Auswertungszeitpunkt die rechte Grenze ist.
Als Ausgabe liefert SEAP zum einen eine graphische Darstellung der Differenzen der Modellsignale vom Referenzsignal in Form von Funktionsgraphen der Differenzfunktionen und zum anderen eine Normierung dieser Funktionen in Form eines Zahlenwerts. Dieser stellt eine Relation der Abweichung der Modellsignale vom Referenzsignal dar und ist somit zur Bewertung ihrer Qualität geeignet.

Berechnung der Distanzen
Nach der Verfeinerung der Signaldaten aus der CSV-Datei kann die Bestimmung der Distanzen zwischen den Referenzsignal und den Modellsignalen durchgeführt werden. Wie bereits erwähnt, sind in SEAP zwei verschiedene Methoden zur Bestimmung dieser Distanzen implementiert: die vertikale und die minimale (Euklidische) Distanz. Im nachfolgenden Abschnitt werden diese anhand von Beispielen erläutert.

Berechnung der vertikalen Distanz:
Es seien zwei Signale f_1 und f_2 über einer Menge von Auswertungszeitpunkten $[t_0,t_1,t_2,t_3,t_4]$ gegeben, wobei f_1 das Referenzsignal und f_2 ein Modellsignal ist (siehe Bild E.5 und E.6). Die vertikale Distanz z_V an der Stelle $t_i \in [t_0,t_1,t_2,t_3,t_4]$ ist definiert als der Absolutbetrag der Diffe-

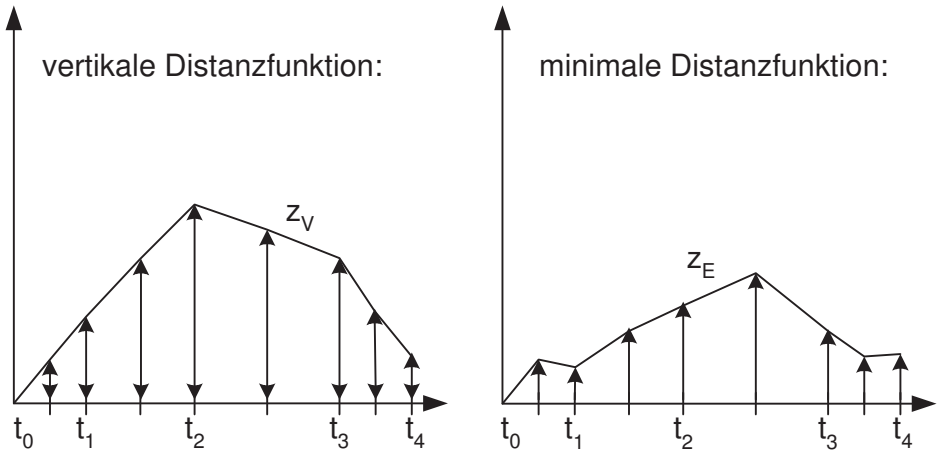

Abbildung E.8: Die vertikale/minimale Distanzfunktion mit Faktor 2

renz der Ordinaten an dieser Stelle t_i.

$$z_V(t_i) = |f_1(t_i) - f_2(t_i)| \tag{E.11}$$

Berechnung der minimalen Distanz:
Die minimale Distanz z_E von f_1 und f_2 an einer Stelle $t_i \in [t_0, t_1, t_2, t_3, t_4]$ ist definiert als minimale Strecke vom Punkt $(t_i, f_1(t_i))$ zu einem beliebigen anderen Punkt $(t, f_2(t))$ auf der Kurve f_2 (siehe Gleichung E.5).

$$z_E(t_i) = \min_{t_i \in [t_0,t_1,t_2,t_3,t_4]} \sqrt{((f_1(t_i) - f_2(t_i))^2 + s^2(t_i - t)^2} \tag{E.12}$$

In Bild E.5 kann man sehen, dass es keine kürzeren Abstände zu Punkten auf der Kurve f_2 gibt als die fett eingezeichneten.
Daraus ergibt sich die minimale Distanzfunktion die in Bild E.6 dargestellt ist.

Anhang F

Liste der Publikationen

1. Mario Anton, Jürgen Weber, Jens Schuster, Andreas Lehmann: Verhaltensmodellierung von Mixed-Signal Automotive ICs für die Anwendung im Virtuellen Test, Analog 2002, Bremen, Mai 2002.

2. Mario Anton, Jürgen Weber: Simulationsmethodik für integrierte Mixed-Signal Automotive Schaltkreise, Analog 2003, Heilbronn, September 2003.

3. Jürgen Weber, Mario Anton und Sorin A. Huss: Verhaltensmodellierung von Ein- und Ausgangsstufen für den Virtuellen Test von Mixed-Signal Automotive Schaltkreisen, Analog 2003, Heilbronn, September 2003.

4. Jürgen Weber, Mario Anton und Sorin A. Huss: Effiziente Mixed-Level Modellierung integrierter Mixed-Signal Automotive Schaltkreise, Analog 2005, Hannover, März 2005.

5. Thomas Markwirth, Peter Schneider, Arno Sörensen, Mario Anton, Jürgen Weber: Verhaltensmodellierung eines adaptiven leistungselektronischen Mixed-Signal-ASIC -> Erweiterung zur mechatronischen Systemsimulation ,GMM/ITG-Diskussionssitzung, Entwicklung von Analogschaltungen mit CAE-Methoden, Analog 2007, Dresden, September 2007.

6. Andreas Lehmler, Jürgen Weber, Mario Anton und Sorin A. Huss: Verifikation der Testschaltung und Simulation der Prüfvorschrift eines Automotive ICs, ASIM, Bremen, März 2007.

7. Jürgen Weber, Andreas Lemke, Mario Anton und Sorin A. Huss: Komponentenbasierte Mixed-Level-Modellierung mit variablen MOSFET-Verhaltensmodellen, ASIM, Bremen, März 2007.

8. Jürgen Weber, Andreas Lemke, Andreas Lehmler, Mario Anton und Sorin A. Huss: Mixed-Level Modeling Using Configurable MOS Transistor Models, FDL'07, Barcelona, September 2007.

9. Jürgen Weber, Andreas Lemke, Andreas Lehmler, Mario Anton und Sorin A. Huss: Mixed-Level Modeling Using Configurable MOS Transistor Models, Buch: Embedded Systems Specification and Design Languages (Kapitel 10), Springer Verlag, 2008.

Literaturverzeichnis

[Abi01] A. A. Abidi: *Behavioral Modeling of Analog and Mixed Signal IC's*, IEEE International Conference, 2001, Seite 443-450.

[Ant99] M. Anton: *Systematischer Einsatz von Least-Squares-Verfahren für die Modellbildung in der Mikrosystemtechnik und Mikroelektronik*,Dissertation, Shaker Verlag, 1999.

[Ant02] M. Anton, J. Weber, J. Schuster, A. Lehmann: *Verhaltensmodellierung von Mixed-Signal Automotive ICs für die Anwendung im Virtuellen Test*, Analog'2002, Bremen 2002, Seite 55-60.

[Ant03] M. Anton, J. Weber: *Simulationsmethodik für integrierte Mixed-Signal Automotive Schaltkreise*, Analog'2003, Heilbronn 2003, Seite 103-108.

[Bab99] B. Babba, G. Barret, F. Poullet: *Virtual Test with VHDL-AMS for a Generator of Analog and Mixed Signal Virtual Components*, IEEE Fall VIUF Workshop, 1999.

[Buc96] M. Bucher, C. Lallement, C. Enz, F. Krummenacher: *Accurate MOS modelling for analog circuit simulation using the EKV model*,Circuits and Systems, 1996. ISCAS '96., 'Connecting the World'., 1996 IEEE International Symposium on Volume 4, 1996, Seite 703-706.

[Bor98] C. Borchers: *Symbolic Behavioral Model Generation of Nonlinear Analog Circuits*, IEEE Transactions on circuits and systems-II, Vol. 45, No. 10, 1998.

[Bor87] I.N. Bronstein, K.A Semendjajew: *Taschenbuch der Mathematik*, Buch Verlag Harri Deutsch, 1987, ISBN 3-87 144-492-8.

[Box87] G. E. P. Box: *Empirical Model Building and Response Surfaces*, John Willey, New York 1987.

[Cad03A] Cadence: *Cadence AMS Environment User Guide*, Cadence Design Systems, Inc. ,Version 5.0, März 2003.

[Cad03B] Cadence: *Cadence AMS Simulator User Guide*, Cadence Design Systems, Inc., Version 5.0, März 2003.

[Cad07] Cadence: *Cadence Device Model Reference*, Cadence Design Systems, Inc., 2007.

[Che99] W.K. Chen: *The VLSI Handbook*, CRC Press LLC, 1999.

[Chr99] E. Christen, K. Bakalar: *VHDL-AMS-a hardware description language for analog and mixed-signal applications*,Circuits and Systems II, Analog and Digital Signal Processing, IEEE Transactions on Volume 46, Issue 10, 1999, Seite 1263-1272.

[Dam97] K. Damm: *Selbsttest integrierter Sensoren in der Betriebsphase mit den Schwerpunkten Signalübertragung und Signalverarbeitung*, Dissertation, Shaker Verlag, 1998.

[Eck00] R. Jancke, S. Zizala, J. Eckmüller. P. Trappe: *Eine Methodik zur Modellierung komplexer Mixed-Signal Baugruppen*, Abschlussworkshop HDL-VMS, Reutlingen 2000, Seite 2/1-2/13.

[Eck99] J. Eckmüller, R. Jancke, A. Schwaferts, P. Schwarz, P. Trappe: *Verhaltensmodellierung und Bibliothekskonzept für analoge Grundschaltungen der Telekommunikation*, Analog'99, München 1999, Seite 40-45.

[Ein95] K. Einwich, P. Schwarz, J. Haase, R. Prescher: *Makromodellierung für Mixed-Signal-Schaltungen*, Mikroelektronik + Mikrosystemtechnik, Heft 4/1995.

[Ein99A] K. Einwich: *DUT-Modellierung für den virtuellen Test*, 11.ITG Workshop Testmethoden und Zuverlässigkeit von Schaltungen und Systemen, Potsdamm-Hermannswerder 1999, Seite 85-89.

[Ein99B] K. Einwich , G. Krampl , R. Hoppenstock , P. Koutsandreas , S.Sattler: *A Multi-Level Modeling Approach rendering Virtual Test Engineering (VTE) Economically Viable for Highly Complex Telecom Circuits*, DATA'99, München 1999, Proceedings User's Forum, Seite 227-231.

[Ein99C] K. Einwich, S. Altmann, T. Leitner, R. Hoppenstock, G. Krampl, S. Sattler: *Applying High-Level Virtual Test to a Complex Mixed-Signal Telecommunication Circuit*, International Mixed-Signal Testing Workshop, Delta Whistler Resort, Vancouver(Whistler), British Columbia, Canada 1999, Seite 91-95.

[Ein98] K. Einwich , P. Schwarz, G. Krampl , P. Trappe , T. Chambers, H. Zojer, S. Sattler: *Virtual test of complex Mixed-Signal Telecommunication Circuits reusing System-level Models*, 4th IEEE International Mixed-Signal Testing Workshop, The Hague, The Netherlands 1998, Seite 237-242.

[Enr03] D. Enright, R.J. Mack, R.E. Massara: *Mixed-Level hierarchical analogue modelling*, Circuits, Devices and Systems, IEE Proceedings Volume 150, Februar 2003, Seite 78-84.

[Esc88] B. Eschermann, W.M. Dai, E.S. Kuh, M. Pedram: *Hierarchical Placement for Macromodels: A Meet-In-The-Middle Approach*, IEEE International Conference, November 1988, Seite 460-463.

[Eva91] C. Evans: *Managing the design of large systems on silicon*, Design Management Environments in CAD, IEE Colloquium on Volume, Januar 1991 Seite 811 - 814.

[Fit03] D. Fitzpatrick, I. Miller: *Analog Behavioral Modeling with the Verilog-A Language*, Buch: Kluwer Academic Publishers, 2003, ISBN 0-7923-8044-4 .

[För99] K. Förster: *VIRTUS als Herausforderung und Chance für flexible und marktgerechte Mixed-Signal-Messtechnikentwicklung aus der Sicht eines ASIC-Entwicklers und -Produzenten*, DATE'99, Berlin 1999.

[Gin02] A. Ginés, E. Peralías, A. Rueda, N. Madrid, R. Seepold: *A Mixed-Signal Design Reuse Methodology Based on Parametric Behavioural Models with Non-Ideal Effects*, DATE'02, Paris 2002, ISBN 0-7695-1471-5.

[Gaj92] D. Gajski, N. Dutt, A. Wu und S. Lin: *High-Level Synthesis: Introduction to Chip and System Design*, Kluwer Academic Publishers, 1992.

[Gei90] R.L. Geiger, P.E. Allen, N.R. Strader: *VLSI Design Techniques for analogand digital circuits*, Buch : McGraw-Hill Publishing Company, 1990, ISBN 0-07-023253-9.

[Gra99A] H. Grams: *Testvorbereitung und Testdebugging mit Virtual Test,* SSE99, Berlin 1999.

[Gra99B] H. Grams, W. Hartl, M. Hayn, W. Tenten: *Virtual Test enables Test Program Verification and Debugging without Tester Hardware*, IMSTW99, Whistler 1999.

[Gra99C] H. Grams, W. Hartl, M. Hayn, W. Tenten: *Virtual Test enables Test Program Verification and Debugging without Tester Hardware*, IMSTW99, Whistler 1999.

[Gras99] A. Grassmann, W. Glauert, M. Miegler: "Novel Simulation Methods as a Solution for Computation Time Problems in Virtual Test", DATE 1999, Berlin 1999.

[Gri96] C. Grimm, K. Waldschmidt: *KIR - A praph-based model for description of mixed analog/digital systems,* Technische Informatik, Universtität Frankfurt 1996.

[Hal02] T. Halfmann: *Analog Insydes: Symbolic Methods in Industrial Analog Circuit Design,*8th International Conference on Applications of Computer Algebra (ACA 2002), Volos Greece 2002.

[Hed03] L. Hedrich: *Structural Synthesis on TransistoR Level: Analysis and Modeling Approaches*, FDL'03, Frankfurt, 2003.

[Heu98A] F. Heuschen, C. Grimm, K. Waldschmidt: *Aspects of Reuse in the Design of Mixed-Signal Systems*, Proceedings 2nd GI/IGT/GMM-Workshop Reuse Techniques for VLSI Design, Karlsruhe 1998.

[Heu98B] F. Heuschen, C. Grimm, K. Waldschmidt: *Top-Down Design of Mixed-Signal Systems with KANDIS*, Proceedings SDA'98, Workshop on System Design Automation, Dresden 1998.

[Hus99] S.A. Huss, S. Klupsch, R. Rosenberger: *Modellierung gemischt analog/digitaler Schaltungen mit VHDL-AMS*, Nachtrag zum Tagungsband, 8. GMM-Workshop Methoden und Werkzeuge zum Entwurf von Mikrosystemen, Berlin 1999, ISSN 0947-1413.

[Hus01A] S.A. Huss: *Model engineering in mixed-signal circuit design*, the KLUWER INTER-NATIONAL SERIES IN ENGINEERING AND COMPUTER SCIENCE 649, Hardbound 2001, ISBN 0-7923-7598-X.

[Hus01B] S.A. Huss, W. John, R. Laur et al.: *Zukünftige Schwerpunkte für die Themen: Entwurf- Simulation-Modellierung-Test*, Arbeitskreis MST-Entwurfstechnik, Dezember 2001, www.vdivde-it.de/mst/entwurf/aktivit/doks/positionspapier.pdf.

[Hus02] S.A. Huss: *VHDL-AMS Modellrepräsentationen und Rollen im Entwurfsablauf für gemischt analog/digitale Schaltungen*, VHDL-AMS Tutorial, ANALOG'02: Bremen 2002.

[IEE93] IEEE VHDL subPAR 1076.1: *Analog Extensions to VHDL*, Design Objective Document, Version 1.1, März 1993.

[Jor01] P. Jores: *Systematic Analog Design Environment'*,BMBF-Abschlussbericht MEDEA A409 Anastasia+, June 2001.

[Kem96] U. Kemper, B. Teutenberg: *Werkzeug zur Generierung funktionaler Beschreibung von Kennlinienverläufen analoger Bauelemente*, Analog'96, Berlin 1996.

[Kle] S. E. Klenke, G. M. Reese: *Modal Test Optimization Using VETO*, http://endo.sandia.gov/9234/papers/veto/matlab.html.

[Klu01] S. Klupsch, S.A. Huss: *Abstraktionsebenen und ihre Anwendung bei der Modellierung und Simulation von heterogenen Systemen*, Vorträge des 3. ITG/GI/GMM-Workshop „Multi-Nature Systems", Hamburg-Harburg 2001, ISBN 3-930400-31-6.

[Kun00] K. Kundert: *A Formal Top-Down Design Process for Mixed-Signal Circuits*, Analog Circuit Design, April 2000.

[Kun04] K. Kundert, O. Zinke: *The Designer's Guide to Verilog AMS*, Kluwer Academic Publishers, Boston 2004, 1. Auflage.

[Ley95] T. Leyendecker, P.Oehler, K. Waldschmidt: *Spezifikation hybrider Systeme*, Bericht 11/95, FB Informatik, Universität Frankfurt 1995.

[Leh07] A. Lehmler, J. Weber, M. Anton und S.A. Huss: *Verifikation der Testschaltung und Simulation der Prüfvorschrift eines Automotive ICs*, ASIM, Bremen, März 2007.

[LRM04] Accellera Verilog Analog Mixed-Signal Group: *Verilog-AMS Language Reference Manual*, Version 2.2, November 2004.

[Mam96] H.-T. Mammen, T. Mager: *Blockorientierte Modellierung eines Abtast-Halte-Verstärkers*", Analog'96, Berlin 1996.

[Mar08] E.S.J. Martens, G.G.E. Gielen: *High-Level Modeling and Synthesis of Analog Integrated Systems*, Buch: Analog Circuits and Signal Processing Series, Springer Verlag, Januar 2008, ISBN 978-1-4020-6801-0.

[Mieg96] M. Miegler, W. Wolz: *Development of Test Programs in a Virtual Test Environment*, IEEE, VLSI Test Symposium, 1996, Seite 99-102.

[Mil00] I. Miller, T.Cassagnes: *Verilog-AMS Eases Mixed Mode Signal Simulation*: Technical Proceedings of the 2000 International Conference on Modeling and Simulation of Microsystems, Seite 305-308, http://www.nsti.org/publ/MSM2000/T31.01.pdf.

[Mos97] V. Moser, H.P. Amann, F. Pellandini: *Behavioural Modelling of Analogue Systems with ABSynth*, Current Issues in Electronic Modeling Vol. 10, Kluwer Academic Publishers, 1997, ISBN 0-7923-9875-0.

[Mue94] K.D. Müller-Glaser, H. Rauch: *Ablaufstrategien und Rechnerunterstützung beim Mikrosystementwurf*, Abschlußbericht des Verbundprojektes „Untersuchungen zum Entwurf von Mikrosystemen", Reihe: Innovationen in der Mikrosystemtechnik, Band 19, November 1994.

[MzB98] V. Meyer zu Bexten, A. Stürmer: *Design Reuse Experiment for Analog Modules*, Proceedings 2nd GI/IGT/GMM-Workshop Reuse Techniques for VLSI Design, Karlsruhe 1998.

[Prä91] H. Prähofer: *System Theoretic Foundations for Combined Discrete-Continuous System Simulation*, Dissertation, Johannes Kepler Universität Linz, Linz 1991.

[Rag93] V. Raghavan, R. A. Rohrer, L.T. Pillage, J. Y. Lee, J. E. Bracken, M.M. Alaybeyi: *AWE-inspired*, in Proc. Custom Integrated Circuits Conf. 1993, Seite 18.1.1-8.

[Rio99] JJ. O. Riordan: *Design of a test simulation environment for test program development*, IEEE 1999, Seite 237-244.

[Rol97] D. Rolince: *Applying Virtual Test Principles to Digital Test Program Development*, IEEE 1997, Seite 72- 75.

[Ron01] M. Rona, G. Krampl: *Modelling SoC Devices for Virtual Test Using VHDL*, Design, Automation and Test in Europe, Conference and Exhibition 2001, Seite 770 -771.

[Ros01] R. Rosenberger: *Zur Generierung von Verhaltensmodellen für gemischt analog/digitale Schaltungen auf Basis der Theorie dynamischer Systeme*, Dissertation, TU Darmstadt, 2001.

[Sax00] E. Sax, J. Willibald: *Abschlußbericht VIRTUS*, FKZ:01 M 3038 E, Informationstechnische Bibliothek Hannover, 2000.

[Sch98] J. Schuster, U. Jäger, I. D. Elliott: *Virtuelle Test-Bench zum optimierten Testen von analogen und Mixed-Signal Ics*, Zeitschrift Horizonte, Oktober 1998.

[Schm98] S. Schmitz, et al.: *A New State-of-the-Art Keyless Entry System*, SAE-Papers, Stuttgart 1998.

[Schw98] P. Schwarz, J. Haase: *Behavioral Modeling of Complex Heterogeneous Microsystems*, FDL'98, Lausanne 1998, Seite 53-62.

[Ste93] U. Steinhausen: *System-Synthesis Using Hardware/Software Codesign*, Int. Workshop on Hardware-Software Co-Design, Cambridge, MA, Oktober 1993

[Som02] R. Sommer, I. Rugen-Herzig, et al.: *From system specification to layout: Seamless topdown design methods for analog and mixed-signal applications*, DATE'02, Paris 2002, ISBN 0-7695-1471-5.

[Tie99] U. Tietze und C. Schenk: *Halbleiter Schaltungstechnik*, Springer Verlag, 1999, 11. Auflage.

[Tri97] R. Trihy: *Analog Extensions to Verilog*, Analog and Mixed-Signal Hardware Description Languages, Current Issues in Electronic Modeling Vol.10, Kluwer Academic Publishers, 1997, ISBN 0-7923-9875-0.

[Udr00] F. Udrea, D. Garner, K. Sheng, A. Popescu, H.T. Lim, V.I. Milne: *SOI power devices*, Electronics and Communication Engineering Journal, Vol. 12, Issue 1, Feb. 2000 Seite 27-40.

[Wal85] R. Walker, D. Thomas: *A Model of Design Representation and Synthesis*, 22nd Design Automation Conference, Las Vegas 1985, ISBN:0-8186-0635-5.

[Wan99] Y. Wang, H.N. Tan: *The development of analog SPICE behavioral model based on IBIS model*, VLSI, 1999. Proceedings. Ninth Great Lakes Symposium, März 1999, Seite: 101-104.

[Web03] J. Weber, M. Anton, S.A. Huss: *Verhaltensmodellierung von Ein- und Ausgangsstufen für den Virtuellen Test von Mixed-Signal Automotive Schaltkreisen*, Analog 2003, Heilbronn 2003, Seite 91-96.

[Web05] J. Weber, M. Anton, S.A. Huss: *Effiziente Mixed-Level Modellierung integrierter Mixed-Signal Automotive Schaltkreise*, Analog 2005, Hannover 2005, Seite 217-222

[Web07A] J. Weber, A. Lemke, M. Anton und S.A. Huss: *Komponentenbasierte Mixed-Level-Modellierung mit variablen MOSFET-Verhaltensmodellen*, ASIM, Bremen, März 2007

[Web07B] J. Weber, A. Lemke, A. Lehmler, M. Anton und S.A. Huss: *Mixed-Level Modeling Using Configurable MOS Transistor Models*, FDL'07, Barcelona, September 2007

[Xun04] Z. Xunyu, C. Hutchens: *EKV model extraction for PD SOI MOSFET* ,Region 5 Conference: Annual Technical and Leadership Workshop, 2004, Seite 81-83.

[Zei84] B.P. Zeigler: *Multifacetted Modelling and Discrete Event Simulation*, Academic Press, 1984.

Abbildungsverzeichnis

Lebenslauf

Persönliche Daten

Name:	Jürgen Weber
Geburtsdatum:	19.10.1973
Geburtsort:	Eberbach
Nationalität:	Deutsch

Schulausbildung

1980-1984	Grundschule Waldbrunn
1984-1990	Realschule Eberbach
	Abschluß: Mittlere Reife
1990-1993	Technisches Gymnasium Mosbach
	Abschluß: Hochschulreife

Wehrdienst

1993-1994	Wehrdienst in Hardheim

Studium

1994-1999	Fachhochschule Heilbronn
	Studiengang: Elektronik
	Abschluß: Dipl.-Ing. (FH)
1999-2001	Fachhochschule Heilbronn und Universtität Newcastle
	Studiengang: Signal Processing and Control
	Abschluß: Master of Science

Beruf

seit 2001	Entwicklungsingenieur
	bei der Firma ATMEL Germany GmbH
seit 2007	Leiter des Teams „Methods and Flows“
	in der CAD Abteilung